



COORDINATED HIGHWAYS ACTION RESPONSE TEAM  
STATE HIGHWAY ADMINISTRATION

---

# **CHART Release 9 Detailed Design**

**Contract SHA-06-CHART  
Document # WO24-DS-001  
Work Order 24, Deliverable 4**

**August 26, 2011  
By  
CSC**





# Table of Contents

---

<b>1</b>	<b>Introduction.....</b>	<b>1-1</b>
1.1	Purpose .....	1-1
1.2	Objectives .....	1-1
1.3	Scope .....	1-1
1.4	Design Process .....	1-2
1.5	Design Tools .....	1-2
1.6	Work Products .....	1-2
<b>2</b>	<b>Architecture .....</b>	<b>2-1</b>
2.1	Network/Hardware.....	2-1
2.2	Software .....	2-1
2.2.1	COTS Products .....	2-1
2.2.1.1	CHART .....	2-1
2.2.2	Deployment /Interface Compatibility .....	2-4
2.2.2.1	CHART .....	2-4
2.3	Security .....	2-8
2.4	Data .....	2-8
2.4.1	Data Storage.....	2-8
2.4.1.1	Database.....	2-8
2.4.1.2	CHART Flat Files .....	2-33
2.4.2	Database Design .....	2-36
2.4.2.1	Decision Support.....	2-36
2.4.2.2	Desktop Video .....	2-36
2.4.2.3	G4 RTMS.....	2-36
2.4.2.4	Archiving - Changes .....	2-37
<b>3</b>	<b>Key Design Concepts .....</b>	<b>3-1</b>
3.1	Decision Support .....	3-1
3.1.1	Key Design Decisions (Decision Support) .....	3-1
3.2	Desktop Video .....	3-2
3.3	G4 RTMS.....	3-6
3.4	Error Processing .....	3-7
3.5	Packaging .....	3-7
3.5.1	CHART.....	3-7
3.6	Assumptions and Constraints .....	3-10
3.6.1	Decision Support .....	3-10
3.6.2	Desktop Video .....	3-10
3.6.3	G4 RTMS .....	3-10

<b>4</b>	<b>Use Cases – Decision Support .....</b>	<b>4-1</b>
4.1.1.1	RespondToTrafficEvent (Use Case Diagram) .....	4-1
4.1.1.2	ConfigureDecisionSupport (Use Case Diagram) .....	4-6
<b>5</b>	<b>Detailed Design – Decision Support.....</b>	<b>5-1</b>
<b>5.1</b>	<b>Human-Machine Interface.....</b>	<b>5-1</b>
5.1.1	Decision Support Features .....	5-1
5.1.1.1	Viewing Suggested DMSs .....	5-1
5.1.1.2	Upstream Devices .....	5-2
5.1.1.3	Other Devices .....	5-4
5.1.1.4	Viewing Additional Information for a Suggested DMS .....	5-5
5.1.1.5	Selecting DMSs and Suggested Messages.....	5-6
5.1.1.6	Adding DMSs to the Response Plan.....	5-7
5.1.1.7	Removing DMSs and Suggested Messages from the Suggestion List.....	5-7
5.1.1.8	Viewing the Suggested Plan DMSs .....	5-8
5.1.1.9	Selecting Plan DMSs .....	5-8
5.1.1.10	Adding Plan DMSs to the Response Plan .....	5-9
5.1.1.11	Removing Plan DMSs from the Suggestion List .....	5-9
5.1.1.12	Getting Suggestions for Additional Recommended DMSs.....	5-9
5.1.1.13	Getting Suggested Messages for DMSs in the Response Plan.....	5-10
5.1.1.14	Warning About DMSs That Are Not Recommended .....	5-11
5.1.1.15	Response Plan Preview Map.....	5-12
5.1.2	Configuring Decision Support Settings .....	5-13
5.1.2.1	General Settings .....	5-14
5.1.2.2	Configuring Distances .....	5-14
5.1.2.3	Configuring Proximity Types .....	5-15
5.1.2.4	Configuring Route Types.....	5-15
5.1.3	Configuring DMS Message Templates.....	5-16
5.1.3.1	Adding a New DMS Message Template.....	5-17
5.1.4	Configuring Word Substitution/Abbreviation List .....	5-21
5.1.5	Configure Event and Incident Type Tag Replacements.....	5-22
5.1.6	Configure Route Direction Replacements .....	5-23
5.1.7	Additional Feature in Response Plan .....	5-24
<b>5.2</b>	<b>Lane Closure Description Generation Flow Chart.....</b>	<b>5-25</b>
<b>5.3</b>	<b>System Interfaces .....</b>	<b>5-29</b>
5.3.1	Class Diagrams .....	5-29
5.3.1.1	Common (Class Diagram) .....	5-29
5.3.1.2	DecisionSupport (Class Diagram) .....	5-35
5.3.1.3	MessageTemplateManagement (Class Diagram).....	5-38
5.3.1.4	TrafficEventManager (Class Diagram) .....	5-41
5.3.1.5	TrafficEventManager2 (Class Diagram) .....	5-46
<b>5.4</b>	<b>TrafficEventModule .....</b>	<b>5-51</b>
5.4.1	Class Diagrams .....	5-51
5.4.1.1	TrafficEventModuleClasses2 (Class Diagram).....	5-51
5.4.2	Sequence Diagrams.....	5-54
5.4.2.1	TrafficEventGroup:disableSuggestionsFor (Sequence Diagram) .....	5-54
5.4.2.2	TrafficEventGroup:enableSuggestionsFor (Sequence Diagram) .....	5-55
5.4.2.3	TrafficEventGroup:requestSuggestions (Sequence Diagram) .....	5-56



5.4.2.4	TrafficEventModule2:initialize (Sequence Diagram)	5-57
<b>5.5</b>	<b>DecisionSupportUtility</b>	<b>5-58</b>
5.5.1	Class Diagrams	5-58
5.5.1.1	DecisionSupportUtility (Class Diagram)	5-58
5.5.2	Sequence Diagrams	5-61
5.5.2.1	DecisionSupportManager:applyTemplate (Sequence Diagram)	5-61
5.5.2.2	DecisionSupportManager:generateSuggestionsForDMS (Sequence Diagram)	5-62
5.5.2.3	DecisionSupportManager:getDMSList (Sequence Diagram)	5-63
<b>5.6</b>	<b>CHART2.Utility.ObjectCache</b>	<b>5-64</b>
5.6.1	Class Diagrams	5-64
5.6.1.1	ObjCachePlanRelatedClasses (Class Diagram)	5-64
5.6.1.2	ObjectCacheDMSMsgTemplateRelatedClasses (Class Diagram)	5-66
<b>5.7</b>	<b>MessageTemplateModule</b>	<b>5-69</b>
5.7.1	Class Diagrams	5-69
5.7.1.1	MessageTemplateModule (Class Diagram)	5-69
5.7.2	Sequence Diagrams	5-73
5.7.2.1	DMSTravInfoMsgTemplateImpl:getConfig (Sequence Diagram)	5-73
5.7.2.2	DMSTravInfoMsgTemplateImpl:setConfig (Sequence Diagram)	5-74
5.7.2.3	MessageTemplateFactoryImpl:createDMSMsgTemplate (Sequence Diagram)	5-75
5.7.2.4	MessageTemplateFactoryImpl:getDMSMsgTemplates (Sequence Diagram)	5-76
<b>5.8</b>	<b>DMSUtility (DMS Templates)</b>	<b>5-77</b>
5.8.1	Class Diagrams	5-77
5.8.1.1	DMSMsgTemplateCommonClasses (Class Diagram)	5-77
5.8.1.2	DMSMsgTemplateDecisionSupprtClasses (Class Diagram)	5-79
5.8.2	Sequence Diagrams	5-83
5.8.2.1	DMSDecSupMsgTemplateModel:formatMulti (Sequence Diagram)	5-83
5.8.2.2	DMSDecSupTemplateRow:formatMultiPrivate (Sequence Diagram)	5-84
5.8.2.3	DMSDecSupTemplateRow:formatMultiPublic (Sequence Diagram)	5-85
5.8.2.4	DSTemplateTag:getMulti (Sequence Diagram)	5-86
<b>5.9</b>	<b>chartlite.data.trafficevents</b>	<b>5-87</b>
5.9.1	Class Diagrams	5-87
5.9.1.1	chartlite.data.trafficevents.DecisionSupport (Class Diagram)	5-87
5.9.1.2	chartlite.data.trafficevents_classes (Class Diagram)	5-89
<b>5.10</b>	<b>chartlite.servlet.templates</b>	<b>5-91</b>
5.10.1	Class Diagrams	5-91
5.10.1.1	GUIMessageTemplateServletClasses (Class Diagram)	5-91
5.10.2	Sequence Diagrams	5-93
5.10.2.1	MessageTemplateReqHdlr:removeDMSDecSupMsgTemplate (Sequence Diagram)	5-93
<b>5.11</b>	<b>chartlite.servlet.trafficevents</b>	<b>5-94</b>
5.11.1	Class Diagrams	5-94
5.11.1.1	chartlite.servlet.trafficevents_classes (Class Diagram)	5-94
5.11.2	Sequence Diagrams	5-96
5.11.2.1	chartlite.servlet.trafficevents.ResponsePlanReqHdlr:addSuggDMSsToResponsePlan (Sequence Diagram)	5-96
5.11.2.2	chartlite.servlet.trafficevents.ResponsePlanReqHdlr:getSuggDMSActions (Sequence Diagram)	5-97
5.11.2.3	chartlite.servlet.trafficevents.ResponsePlanReqHdlr:viewSuggActionsCommandStatus	

(Sequence Diagram) .....	5-98
5.11.2.4 chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:viewSuggDMSActions (Sequence Diagram) .....	5-99
5.11.2.5 ResponsePlanReqHdlr:getResponsePlanDecSuppData (Sequence Diagram) .....	5-100
5.11.2.6 ResponsePlanReqHdlr:viewResponsePlanPreviewMap (Sequence Diagram) .....	5-101
<b>5.12 chartlite.servlet.dms .....</b>	<b>5-102</b>
5.12.1 Class Diagrams .....	5-102
5.12.1.1 GUIDMSServletClasses (Class Diagram) .....	5-102
5.12.2 Sequence Diagrams .....	5-104
5.12.2.1 chartlite.servlet.dms.DMSReqHdlr:addDMSDecSuppMsgTemplate (Sequence Diagram) .....	5-104
5.12.2.2 chartlite.servlet.dms.DMSReqHdlr:editDMSDecSuppMsgTemplate (Sequence Diagram) .....	5-105
<b>5.13 chartlite.servlet.usermgmt .....</b>	<b>5-106</b>
5.13.1 Class Diagrams .....	5-106
5.13.1.1 chartlite.servlet.usermgmt.systemProfile_classes (Class Diagram) .....	5-106
5.13.2 Sequence Diagrams .....	5-108
5.13.2.1 SystemProfileReqHdlr:getDecisionSupportGeneralSettingsForm (Sequence Diagram) .....	5-108
5.13.2.2 SystemProfileReqHdlr:setDecisionSupportGeneralSettings (Sequence Diagram) .....	5-109
<b>5.14 Map Views (javascript) .....</b>	<b>5-110</b>
5.14.1 Class Diagrams .....	5-110
5.14.1.1 MapViewSpecificClasses (Class Diagram) .....	5-110
5.14.2 Sequence Diagrams .....	5-112
5.14.2.1 ResponsePlanPreviewMap:handleMapDataJSON (Sequence Diagram) .....	5-112
5.14.2.2 ResponsePlanPreviewMap:initialize (Sequence Diagram) .....	5-113
<b>6 Use Cases – Desktop Video .....</b>	<b>6-1</b>
<b>6.1 R9VideoOnDesktopUses (Use Case Diagram) .....</b>	<b>6-1</b>
6.1.1 Administrator (Actor) .....	6-1
6.1.2 Block Camera Display To Flash Server (Use Case) .....	6-1
6.1.3 Block Camera Display To Public (Use Case) .....	6-2
6.1.4 Cancel Video Session For User (Use Case) .....	6-2
6.1.5 Configure Flash Server (Use Case) .....	6-2
6.1.6 Configure Flash Streaming Control (Use Case) .....	6-2
6.1.7 Configure Operations Center (Use Case) .....	6-2
6.1.8 Export Camera Info (Use Case) .....	6-2
6.1.9 Monitor Video Sessions (Use Case) .....	6-2
6.1.10 Override Camera Control (Use Case) .....	6-3
6.1.11 Release Camera Control (Use Case) .....	6-3
6.1.12 Request Camera Control (Use Case) .....	6-3
6.1.13 System (Actor) .....	6-3
6.1.14 Unblock Camera Display To Public (Use Case) .....	6-3
6.1.15 Unblock Camera Display From Flash Server (Use Case) .....	6-4
6.1.16 User (Actor) .....	6-4
6.1.17 View Operations Center List (Use Case) .....	6-4

6.1.18	View Video And Camera Control Dialog (Use Case) .....	6-4
6.1.19	View Desktop Video (Use Case) .....	6-4
6.1.20	View Desktop Video For Source (Use Case).....	6-5
6.1.21	View Desktop Video Tour (Use Case) .....	6-5
6.1.22	View Tour Details (Use Case) .....	6-5
6.1.23	View Video Sessions (Use Case).....	6-5
6.1.24	View Video Source Details (Use Case) .....	6-5
6.1.25	View Video Source List (Use Case) .....	6-6
<b>6.2</b>	<b>R9ViewDesktopVideo (Use Case Diagram) .....</b>	<b>6-7</b>
6.2.1	Acquire Video Session (Use Case) .....	6-7
6.2.2	End Video Session (Use Case) .....	6-7
6.2.3	Monitor Video Source And Session Status (Use Case) .....	6-7
6.2.4	Pause Desktop Video (Use Case) .....	6-8
6.2.5	Play Desktop Video (Use Case).....	6-8
6.2.6	Resize Desktop Video (Use Case) .....	6-8
6.2.7	System (Actor).....	6-8
6.2.8	User (Actor) .....	6-8
<b>7</b>	<b>Detailed Design – Desktop Video .....</b>	<b>7-1</b>
<b>7.1</b>	<b>Human-Machine Interface.....</b>	<b>7-1</b>
7.1.1	Video Sources .....	7-1
7.1.2	Camera / Video Source Details.....	7-3
7.1.3	Tour Details .....	7-5
7.1.4	Video Popup (new) .....	7-6
7.1.5	Camera Control.....	7-9
7.1.6	Camera Map Popup .....	7-13
7.1.7	Operations Centers.....	7-14
7.1.8	Operations Center Details .....	7-15
7.1.9	Video Administration .....	7-16
7.1.10	Video Sessions (new).....	7-17
7.1.11	Streaming Flash Server Configuration.....	7-18
7.1.12	Add Operations Center .....	7-19
7.1.13	Operations Center Settings .....	7-20
<b>7.2</b>	<b>System Interfaces .....</b>	<b>7-21</b>
7.2.1	Class Diagrams .....	7-21
7.2.1.1	ResourceManagement (Class Diagram).....	7-21
7.2.1.2	CameraControlIDLCclasses (Class Diagram) .....	7-25
<b>7.3</b>	<b>ResourcesModule.....</b>	<b>7-34</b>

7.3.1	Class Diagrams .....	7-34
7.3.1.1	ResourceClasses (Class Diagram) .....	7-34
7.3.2	Sequence Diagrams.....	7-37
7.3.2.1	OperationsCenterImpl:endVideoSession (Sequence Diagram) .....	7-37
7.3.2.2	OperationsCenterImpl:requestVideoSession (Sequence Diagram).....	7-38
7.3.2.3	OperationsCenterImpl:touchVideoSessions (Sequence Diagram).....	7-39
7.3.2.4	VideoSessionCleanupTask:run (Sequence Diagram) .....	7-40
<b>7.4</b>	<b>CameraControlModule .....</b>	<b>7-41</b>
7.4.1	State Diagram .....	7-41
7.4.1.1	CameraControlModule (State Diagram) .....	7-41
7.4.2	Class Diagrams .....	7-42
7.4.2.1	CameraControlModule (Class Diagram) .....	7-42
7.4.3	Sequence Diagrams.....	7-53
7.4.3.1	CameraControlModule:BlockToPublic (Sequence Diagram).....	7-53
7.4.3.2	VideoCameraImpl:setSFSBlocked (Sequence Diagram).....	7-55
7.4.3.3	VideoCameraImpl:setStreamsBlockedInPublicSFSs (SD) .....	7-57
7.4.3.4	VideoCameraImpl:unlockToPublic (Sequence Diagram).....	7-58
<b>7.5</b>	<b>chartlite.data .....</b>	<b>7-59</b>
7.5.1	Class Diagrams .....	7-59
7.5.1.1	MiscDataClasses (Class Diagram).....	7-59
7.5.2	Sequence Diagrams.....	7-62
7.5.2.1	ResourceMgmtPushConsumer:handleVideoSessionEnded (Sequence Diagram) .....	7-62
7.5.2.2	WebVideoSession:getWebVideoSessions (Sequence Diagram) .....	7-63
<b>7.6</b>	<b>chartlite.data.video-data .....</b>	<b>7-64</b>
7.6.1	Class Diagrams .....	7-64
7.6.1.1	GUIVideoDataClasses (Class Diagram) .....	7-64
<b>7.7</b>	<b>chartlite.servlet.....</b>	<b>7-66</b>
7.7.1	Class Diagrams .....	7-66
7.7.1.1	ServletBaseClasses (Class Diagram) .....	7-66
<b>7.8</b>	<b>chartlite.servlet.video.....</b>	<b>7-69</b>
7.8.1	Class Diagrams .....	7-69
7.8.1.1	GUIVideoServletClasses (Class Diagram) .....	7-69
7.8.1.2	GUIVideoServletClasses2 (Class Diagram) .....	7-72
7.8.2	Sequence Diagrams.....	7-74
7.8.2.1	ControlVideoSourceReqHdlr:processSetSFSBlocked (Sequence Diagram) .....	7-74
7.8.2.2	TouchVideoSessionsTask:run (Sequence Diagram) .....	7-75
7.8.2.3	VideoAndControlData:getVideoSession (Sequence Diagram).....	7-76
7.8.2.4	VideoAndControlData:releaseVideoSession (Sequence Diagram) .....	7-77
7.8.2.5	VideoAndControlData:requestControlSession (Sequence Diagram).....	7-78
7.8.2.6	VideoAndControlData:requestVideoSession (Sequence Diagram) .....	7-79
7.8.2.7	VideoAndControlData:videoRequiredForControl (Sequence Diagram) .....	7-80
7.8.2.8	VideoAndControlReqHdlr:getVideoAndControlPage (Sequence Diagram).....	7-81
7.8.2.9	VideoAndControlReqHdlr:initVideoAndControlDataJSON (Sequence Diagram) .....	7-82
7.8.2.10	VideoSessionReqHdlr:endOwnVideoSession (Sequence Diagram).....	7-83
7.8.2.11	VideoSessionReqHdlr:endUserVideoSession (Sequence Diagram).....	7-84
7.8.2.12	VideoSessionReqHdlr:updateVideoSessionXML (Sequence Diagram).....	7-85
7.8.2.13	VideoSessionReqHdlr:viewVideoSession (Sequence Diagram) .....	7-86
7.8.2.14	VideoSessionReqHdlr:viewVideoSessionList (Sequence Diagram) .....	7-87

7.8.2.15	VideoTourReqHdlr:viewTourListDetails (Sequence Diagram)	7-89
7.8.2.16	ViewVideoSourceReqHdlr:processViewDetailsReq (Sequence Diagram)	7-90
7.8.2.17	ViewVideoSourceReqHdlr:processViewVideoSourceListReq (Sequence Diagram)	7-92
<b>7.9</b>	<b>video-flex</b>	<b>7-93</b>
7.9.1	Class Diagrams	7-93
7.9.1.1	video_flex_classes (Class Diagram)	7-93
7.9.2	Sequence Diagrams	7-94
7.9.2.1	VideoDisplayApp:init (Sequence Diagram)	7-94
7.9.2.2	VideoDisplayComponent:displayNextEligibleVideoSourceInTour (Sequence Diagram)	7-95
7.9.2.3	VideoDisplayComponent:handleTourTimer (Sequence Diagram)	7-96
7.9.2.4	VideoDisplayComponent:handleUpdateSessionResponse (Sequence Diagram)	7-97
7.9.2.5	VideoDisplayComponent:playAndPause (Sequence Diagram)	7-98
<b>8</b>	<b>Use Cases – G4 RTMS</b>	<b>8-1</b>
<b>8.1</b>	<b>ConfigureTSS (Use Case Diagram)</b>	<b>8-1</b>
8.1.1	Add TSS (Use Case)	8-1
8.1.2	Change TSS Model (Use Case)	8-2
8.1.3	Configure TSS Zone Groups (Use Case)	8-2
8.1.4	Copy TSS (Use Case)	8-2
8.1.5	Edit Map Display Options (Use Case)	8-2
8.1.6	Remove TSS (Use Case)	8-2
8.1.7	Set Basic Serial Communication Settings (Use Case)	8-2
8.1.8	Set Bearing (Use Case)	8-2
8.1.9	Set Device Location (Use Case)	8-3
8.1.10	Set Direct CommunicationSettings (Use Case)	8-3
8.1.11	Set Modem CommunicationSettings (Use Case)	8-3
8.1.12	Set TCPIP Communication Settings (Use Case)	8-3
8.1.13	Set TSS Communication Settings (Use Case)	8-3
8.1.14	Set TSS Configuration (Use Case)	8-3
8.1.15	Set Zone Group Display Direction (Use Case)	8-4
8.1.16	Set Zone Group Display Order (Use Case)	8-4
8.1.17	Specify TSS Model (Use Case)	8-4
<b>8.2</b>	<b>ManageTSS (Use Case Diagram)</b>	<b>8-5</b>
8.2.1	Archive TSS Data (Use Case)	8-5
8.2.2	Create Alert (Use Case)	8-5
8.2.3	Edit TSS System Profile (Use Case)	8-5
8.2.4	Get TSS Status (Use Case)	8-6
8.2.5	Load TSS Data (Use Case)	8-6
8.2.6	Poll TSS (Use Case)	8-6
8.2.7	Put TSS in Maint Mode (Use Case)	8-6

8.2.8	Put TSS Online (Use Case).....	8-6
8.2.9	Run TSS BIT (Use Case).....	8-6
8.2.10	Take TSS Offline (Use Case) .....	8-6
<b>8.3</b>	<b>ViewTSS (Use Case Diagram).....</b>	<b>8-7</b>
8.3.1	Filter List (Use Case).....	8-7
8.3.2	Select List Columns (Use Case) .....	8-7
8.3.3	Sort List (Use Case).....	8-7
8.3.4	View TSS Details (Use Case).....	8-7
8.3.5	View TSS List (Use Case).....	8-8
8.3.6	View TSS Traffic Data (Use Case).....	8-8

## **9 Detailed Design – G4 RTMS .....9-1**

<b>9.1</b>	<b>Human-Machine Interface.....</b>	<b>9-1</b>
9.1.1	Add TSS .....	9-2
9.1.2	Configure TSS .....	9-3
9.1.2.1	Configure Zone Groups .....	9-3
9.1.2.2	Specify Whether to Run Scheduled BIT.....	9-3
9.1.2.3	Change TSS Model Type.....	9-5
9.1.3	View TSS.....	9-7
9.1.4	Manage TSS.....	9-10
<b>9.2</b>	<b>System Interfaces .....</b>	<b>9-11</b>
9.2.1	Class Diagrams .....	9-11
9.2.1.1	TSSManagement (Class Diagram).....	9-11
<b>9.3</b>	<b>TSSManagementPkg .....</b>	<b>9-20</b>
9.3.1	Class Diagrams .....	9-20
9.3.1.1	TSSManagementModulePkg (Class Diagram) .....	9-20
9.3.2	Sequence Diagrams.....	9-28
9.3.2.1	PolledTSSImpl:changeModelType (Sequence Diagram) .....	9-28
9.3.2.2	PolledTSSImpl:processPollResults (Sequence Diagram) .....	9-29
9.3.2.3	RTMSFactoryImpl:constructor (Sequence Diagram) .....	9-30
9.3.2.4	RTMSFactoryImpl:createRTMS (Sequence Diagram).....	9-31
9.3.2.5	RTMSFactoryImpl:CurrentStatusPush (Sequence Diagram) .....	9-32
9.3.2.6	RTMSFactoryImpl:pollDevices (Sequence Diagram) .....	9-33
9.3.2.7	RTMSImpl:runBIT (Sequence Diagram) .....	9-34
9.3.2.8	RTMSImpl:poll (Sequence Diagram).....	9-35
<b>9.4</b>	<b>chartlite.data.tss-data .....</b>	<b>9-37</b>
9.4.1	Class Diagrams .....	9-37
9.4.1.1	GUITSSDataClasses (Class Diagram).....	9-37
<b>9.5</b>	<b>chartlite.servlet.tss .....</b>	<b>9-39</b>
9.5.1	Class Diagrams .....	9-39
9.5.1.1	chartlite.servlet.tss_classes (Class Diagram) .....	9-39
9.5.1.2	chartlite.servlet.tss_dynlist_classes (Class Diagram).....	9-41
9.5.2	Sequence Diagrams.....	9-43

9.5.2.1	chartlite.servlet.tss:parseBasicConfigSettings (Sequence Diagram).....	9-43
9.5.2.2	TSSListSupporter:createDynList (Sequence Diagram) .....	9-44
9.5.2.3	TSSReqHdlr:processAddTSS (Sequence Diagram) .....	9-46
9.5.2.4	TSSReqHdlr:processChangeTSSModel (Sequence Diagram).....	9-47
9.5.2.5	TSSReqHdlr:processRunBIT (Sequence Diagram) .....	9-48
<b>10</b>	<b>Deprecated Functionalities .....</b>	<b>10-1</b>
<b>11</b>	<b>Mapping To Requirements .....</b>	<b>11-1</b>
<b>12</b>	<b>Acronyms/Glossary .....</b>	<b>12-1</b>

## Table of Figures

Figure 2-1 CHART and External Interfaces .....	2-5
Figure 2-2 R9 Server Deployment.....	2-6
Figure 2-3 R9 GUI Deployment.....	2-7
Figure 2-4 R9 ERD.....	2-22
Figure 4-1. RespondToTrafficEvent (Use Case Diagram) .....	4-1
Figure 4-2. ConfigureDecisionSupport (Use Case Diagram) .....	4-6
Figure 5-1. Traffic Event Response Plan Showing Features of Decision Support. ....	5-1
Figure 5-2. Requesting Suggested DMSs for a Traffic Event Response Plan. ....	5-2
Figure 5-3. Suggested Upstream Devices and Messages for a Traffic Event Response Plan. ....	5-3
Figure 5-4. Additional Suggested Messages for a Suggested DMS. ....	5-4
Figure 5-5. The Link to View the Details of Other Devices. ....	5-4
Figure 5-6. Suggested Other Devices and Messages for a Traffic Event Response Plan. ....	5-5
Figure 5-7. DMS Current Message Preview on the Suggested Response Actions Page. ....	5-5
Figure 5-8. Suggested Message Scoring Details.....	5-6
Figure 5-9. Selecting Devices to be Added to the Response Plan. ....	5-6
Figure 5-10. Buttons for Adding and Executing DMSs in the Response Plan. ....	5-7
Figure 5-11. Selected Devices Added to the Response Plan with Suggested Messages.....	5-7
Figure 5-12. The Button for Removing DMSs and Suggested Messages from the Suggestions List. ....	5-7
Figure 5-13. Suggested Plans for a Traffic Event Response Plan.....	5-8
Figure 5-14. Selecting Plan Devices to be Added to the Response Plan. ....	5-9
Figure 5-15. The Link to View Additional Recommended DMSs. ....	5-10
Figure 5-16. Additional Recommended DMSs Suggested for Response Plan. ....	5-10
Figure 5-17. Getting Suggested Messages for DMSs in the Response Plan. ....	5-10
Figure 5-18. Suggested Messages for All DMSs in the Response Plan.....	5-11
Figure 5-19. Response Plan Showing Warning About a DMS That is Not Recommended. ....	5-11
Figure 5-20. The Button for Viewing a Preview of the Response Plan on a Map. ....	5-12
Figure 5-21. The Response Plan Preview Map Showing Recommended and Not Recommended DMSs. ....	5-12
Figure 5-22. The Response Plan Preview Map Layer Switcher Showing Recommended and Not Recommended DMS Layers.....	5-13
Figure 5-23. The Configuring Decision Support Settings Page.....	5-14
Figure 5-24. The Distances Settings Form Showing the Maximum Distances and the Percentage of Lanes Closed. .	5-15
Figure 5-25. The Proximity Settings Showing the Proximities for Which to Suggest Messages. ....	5-15
Figure 5-26. The Route Types Settings Showing the Route Types for Which to Suggest Messages. ....	5-16
Figure 5-27. The Decision Support DMS Message Template List Page. ....	5-17
Figure 5-28. The DMS Message Template Editor with the Tag Selector Popup Displayed.....	5-18
Figure 5-29. The DMS Message Template Editor With All Values Set.....	5-20
Figure 5-30. DMS Message Template Editor With Warning About Message Too Long for Geometry. ....	5-21
Figure 5-31. The Decision Support Word Substitution / Abbreviation List. ....	5-22
Figure 5-32. Adding a New Word Substitution / Abbreviation for Decision Support.....	5-22
Figure 5-33. Configuring Replacement Text for the Traffic Event Type and Incident Type Tags. ....	5-23
Figure 5-34. Configuring Replacement Text for the Valid Roadway Directions and Additional Directions. ....	5-24
Figure 5-35. DMS Current Message Preview on the Traffic Event Details Page.....	5-24
Figure 5-36 Lane Closure Description Main Logic.....	5-25
Figure 5-37 Travel Lanes Affected Logic .....	5-26
Figure 5-38 Exits Lanes Affected Logic.....	5-27
Figure 5-39 Shoulders Affected Logic .....	5-28
Figure 5-40. Common (Class Diagram) .....	5-29
Figure 5-41. DecisionSupport (Class Diagram).....	5-35
Figure 5-42. MessageTemplateManagement (Class Diagram).....	5-38
Figure 5-43. TrafficEventManager (Class Diagram) .....	5-41
Figure 5-44. TrafficEventManager2 (Class Diagram) .....	5-46
Figure 5-45. TrafficEventModuleClasses2 (Class Diagram).....	5-51



Figure 5-46. TrafficEventGroup:disableSuggestionsFor (Sequence Diagram) .....	5-54
Figure 5-47. TrafficEventGroup:enableSuggestionsFor (Sequence Diagram) .....	5-55
Figure 5-48. TrafficEventGroup:requestSuggestions (Sequence Diagram) .....	5-56
Figure 5-49. TrafficEventModule2:initialize (Sequence Diagram) .....	5-57
Figure 5-50. DecisionSupportUtility (Class Diagram) .....	5-58
Figure 5-51. DecisionSupportManager:applyTemplate (Sequence Diagram) .....	5-61
Figure 5-52. DecisionSupportManager:generateSuggestionsForDMS (Sequence Diagram) .....	5-62
Figure 5-53. DecisionSupportManager:getDMSList (Sequence Diagram) .....	5-63
Figure 5-54. ObjCachePlanRelatedClasses (Class Diagram) .....	5-64
Figure 5-55. ObjectCacheDMSMsgTemplateRelatedClasses (Class Diagram) .....	5-66
Figure 5-56. MessageTemplateModule (Class Diagram) .....	5-69
Figure 5-57. DMSTravInfoMsgTemplateImpl:getConfig (Sequence Diagram) .....	5-73
Figure 5-58. DMSTravInfoMsgTemplateImpl:setConfig (Sequence Diagram) .....	5-74
Figure 5-59. MessageTemplateFactoryImpl:createDMSMsgTemplate (Sequence Diagram) .....	5-75
Figure 5-60. MessageTemplateFactoryImpl:getDMSMsgTemplates (Sequence Diagram) .....	5-76
Figure 5-61. DMSMsgTemplateCommonClasses (Class Diagram) .....	5-77
Figure 5-62. DMSMsgTemplateDecisionSupprtClasses (Class Diagram) .....	5-79
Figure 5-63. DMSDecSupMsgTemplateModel:formatMulti (Sequence Diagram) .....	5-83
Figure 5-64. DMSDecSupTemplateRow:formatMultiPrivate (Sequence Diagram) .....	5-84
Figure 5-65. DMSDecSupTemplateRow:formatMultiPublic (Sequence Diagram) .....	5-85
Figure 5-66. DSTemplateTag:getMulti (Sequence Diagram) .....	5-86
Figure 5-67. chartlite.data.trafficevents.DecisionSupport (Class Diagram) .....	5-87
Figure 5-68. chartlite.data.trafficevents_classes (Class Diagram) .....	5-89
Figure 5-69. GUIMessageTemplateServletClasses (Class Diagram) .....	5-91
Figure 5-70. MessageTemplateReqHdlr:removeDMSDecSupMsgTemplate (Sequence Diagram) .....	5-93
Figure 5-71. chartlite.servlet.trafficevents_classes (Class Diagram) .....	5-94
Figure 5-72. chartlite.servlet.trafficevents.ResponsePlanReqHdlr:addSuggDMSsToResponsePlan (Sequence Diagram) .....	5-96
Figure 5-73. chartlite.servlet.trafficevents.ResponsePlanReqHdlr:getSuggDMSActions (Sequence Diagram) .....	5-97
Figure 5-74. chartlite.servlet.trafficevents.ResponsePlanReqHdlr:viewSuggActionsCommandStatus (Sequence Diagram) .....	5-98
Figure 5-75. chartlite.servlet.trafficevents.ResponsePlanReqHdlr:viewSuggDMSActions (Sequence Diagram) ..	5-99
Figure 5-76. ResponsePlanReqHdlr:getResponsePlanDecSupData (Sequence Diagram) .....	5-100
Figure 5-77. ResponsePlanReqHdlr:viewResponsePlanPreviewMap (Sequence Diagram) .....	5-101
Figure 5-78. GUIDMSServletClasses (Class Diagram) .....	5-102
Figure 5-79. chartlite.servlet.dms.DMSReqHdlr:addDMSDecSupMsgTemplate (Sequence Diagram) .....	5-104
Figure 5-80. chartlite.servlet.dms.DMSReqHdlr:editDMSDecSupMsgTemplate (Sequence Diagram) .....	5-105
Figure 5-81. chartlite.servlet.usermgmt.systemProfile_classes (Class Diagram) .....	5-106
Figure 5-82. SystemProfileReqHdlr:getDecisionSupportGeneralSettingsForm (Sequence Diagram) .....	5-108
Figure 5-83. SystemProfileReqHdlr:setDecisionSupportGeneralSettings (Sequence Diagram) .....	5-109
Figure 5-84. MapViewSpecificClasses (Class Diagram) .....	5-110
Figure 5-85. ResponsePlanPreviewMap:handleMapDataJSON (Sequence Diagram) .....	5-112
Figure 5-86. ResponsePlanPreviewMap:initialize (Sequence Diagram) .....	5-113
Figure 6-1. R9VideoOnDesktopUses (Use Case Diagram) .....	6-1
Figure 6-2. R9ViewDesktopVideo (Use Case Diagram) .....	6-7
Figure 7-1 Video Sources: Cameras .....	7-1
Figure 7-2 Video Sources: Tours and Other Sources .....	7-2
Figure 7-3 Video Source Details .....	7-3
Figure 7-4 Tour Details .....	7-5
Figure 7-5 Video Popup (Video Source / Non-Controllable Camera) .....	7-6
Figure 7-6 Video and Control Window (Video Only Mode) .....	7-7
Figure 7-7 Video Popup (Video Tour) .....	7-8
Figure 7-8 Video and Control Window (Control Only Mode) .....	7-10
Figure 7-9 Video and Control Window (Video and Control Mode) .....	7-11
Figure 7-10 Camera Map Popup .....	7-13
Figure 7-11 Operations Centers .....	7-14

Figure 7-12 Operations Center Details .....	7-15
Figure 7-13 Video Administration.....	7-16
Figure 7-14 Video Sessions .....	7-17
Figure 7-15 SFS Configuration .....	7-18
Figure 7-16 Add Operations Center.....	7-19
Figure 7-17 Operations Center Settings.....	7-20
Figure 7-18. ResourceManagement (Class Diagram).....	7-21
Figure 7-19. CameraControlIDLClasses (Class Diagram) .....	7-25
Figure 7-20. ResourceClasses (Class Diagram).....	7-34
Figure 7-21. OperationsCenterImpl:endVideoSession (Sequence Diagram) .....	7-37
Figure 7-22. OperationsCenterImpl:requestVideoSession (Sequence Diagram).....	7-38
Figure 7-23. OperationsCenterImpl:touchVideoSessions (Sequence Diagram).....	7-39
Figure 7-24. VideoSessionCleanupTask:run (Sequence Diagram) .....	7-40
Figure 7-25. CameraControlModule (State Diagram) .....	7-41
Figure 7-26. CameraControlModule (Class Diagram) .....	7-42
Figure 7-27. CameraControlModule:BlockToPublic (Sequence Diagram).....	7-54
Figure 7-28. VideoCameraImpl:setSFSBlocked (Sequence Diagram) .....	7-56
Figure 7-29. VideoCameraImpl:setStreamsBlockedInPublicSFSs (Sequence Diagram).....	7-57
Figure 7-30. VideoCameraImpl:unlockToPublic (Sequence Diagram).....	7-58
Figure 7-31. MiscDataClasses (Class Diagram).....	7-59
Figure 7-32. ResourceMgmtPushConsumer:handleVideoSessionEnded (Sequence Diagram) .....	7-62
Figure 7-33. WebVideoSession:getWebVideoSessions (Sequence Diagram).....	7-63
Figure 7-34. GUIVideoDataClasses (Class Diagram) .....	7-64
Figure 7-35. ServletBaseClasses (Class Diagram) .....	7-66
Figure 7-36. GUIVideoServletClasses (Class Diagram) .....	7-69
Figure 7-37. GUIVideoServletClasses2 (Class Diagram) .....	7-72
Figure 7-38. ControlVideoSourceReqHdlr:processSetSFSBlocked (Sequence Diagram) .....	7-74
Figure 7-39. TouchVideoSessionsTask:run (Sequence Diagram) .....	7-75
Figure 7-40. VideoAndControlData:getVideoSession (Sequence Diagram).....	7-76
Figure 7-41. VideoAndControlData:releaseVideoSession (Sequence Diagram).....	7-77
Figure 7-42. VideoAndControlData:requestControlSession (Sequence Diagram).....	7-78
Figure 7-43. VideoAndControlData:requestVideoSession (Sequence Diagram) .....	7-79
Figure 7-44. VideoAndControlData:videoRequiredForControl (Sequence Diagram) .....	7-80
Figure 7-45. VideoAndControlReqHdlr:getVideoAndControlPage (Sequence Diagram) .....	7-81
Figure 7-46. VideoAndControlReqHdlr:initVideoAndControlDataJSON (Sequence Diagram) .....	7-82
Figure 7-47. VideoSessionReqHdlr:endOwnVideoSession (Sequence Diagram).....	7-83
Figure 7-48. VideoSessionReqHdlr:endUserVideoSession (Sequence Diagram) .....	7-84
Figure 7-49. VideoSessionReqHdlr:updateVideoSessionXML (Sequence Diagram).....	7-85
Figure 7-50. VideoSessionReqHdlr:viewVideoSession (Sequence Diagram) .....	7-86
Figure 7-51. VideoSessionReqHdlr:viewVideoSessionList (Sequence Diagram) .....	7-88
Figure 7-52. VideoTourReqHdlr:viewTourListDetails (Sequence Diagram).....	7-89
Figure 7-53. ViewVideoSourceReqHdlr:processViewDetailsReq (Sequence Diagram) .....	7-91
Figure 7-54. ViewVideoSourceReqHdlr:processViewVideoSourceListReq (Sequence Diagram) .....	7-92
Figure 7-55. video_flex_classes (Class Diagram) .....	7-93
Figure 7-56. VideoDisplayApp:init (Sequence Diagram) .....	7-94
Figure 7-57. VideoDisplayComponent:displayNextEligibleVideoSourceInTour (Sequence Diagram).....	7-95
Figure 7-58. VideoDisplayComponent:handleTourTimer (Sequence Diagram) .....	7-96
Figure 7-59. VideoDisplayComponent:handleUpdateSessionResponse (Sequence Diagram).....	7-97
Figure 7-60. VideoDisplayComponent:playAndPause (Sequence Diagram) .....	7-98
Figure 8-1. ConfigureTSS (Use Case Diagram) .....	8-1
Figure 8-2. ManageTSS (Use Case Diagram) .....	8-5
Figure 8-3. ViewTSS (Use Case Diagram) .....	8-7
Figure 9-1 Add TSS Form .....	9-2
Figure 9-2. TSSManagement (Class Diagram).....	9-12
Figure 9-3. TSSManagementModulePkg (Class Diagram) .....	9-21
Figure 9-4. PolledTSSImpl:changeModelType (Sequence Diagram) .....	9-28

Figure 9-5. PolledTSSImpl:processPollResults (Sequence Diagram) .....	9-29
Figure 9-6. RTMSFactoryImpl:constructor (Sequence Diagram) .....	9-30
Figure 9-7. RTMSFactoryImpl:createRTMS (Sequence Diagram).....	9-31
Figure 9-8. RTMSFactoryImpl:CurrentStatusPush (Sequence Diagram).....	9-32
Figure 9-9. RTMSFactoryImpl:pollDevices (Sequence Diagram) .....	9-33
Figure 9-10. RTMSImpl:runBIT (Sequence Diagram).....	9-34
Figure 9-11. RTMSImpl:poll (Sequence Diagram) .....	9-36
Figure 9-13. GUITSSDataClasses (Class Diagram) .....	9-37
Figure 9-14. chartlite.servlet.tss_classes (Class Diagram) .....	9-39
Figure 9-15. chartlite.servlet.tss_dynlist_classes (Class Diagram).....	9-41
Figure 9-16. chartlite.servlet.tss:parseBasicConfigSettings (Sequence Diagram).....	9-43
Figure 9-17. TSSListSupporter:createDynList (Sequence Diagram) .....	9-45
Figure 9-18. TSSReqHdlr:processAddTSS (Sequence Diagram) .....	9-46
Figure 9-19. TSSReqHdlr:processChangeTSSModel (Sequence Diagram).....	9-47
Figure 9-20. TSSReqHdlr:processRunBIT (Sequence Diagram) .....	9-48

# 1 Introduction

---

## 1.1 Purpose

This document describes the design of the software for CHART Release. This build provides:

- Decision Support – CHART R9 will, on user demand, generate a suggestion of the best DMS devices and messages to use in response to a traffic event. Nearby DMSs are included based on proximity rules and the type and severity of the traffic event, and messages are generated using pre-configured templates which allow substitution of traffic event parameters into the message.
- Desktop Video: CHART R9 will allow users to display video from cameras and tours in their web browser windows. Users will be able to display video in a standalone window or (for controllable cameras) within the Camera Control window. To limit bandwidth usage, video sessions will be limited in number per operations center, and tracked to show who is viewing the video for a given camera or tour. A user with sufficient privileges will be able to remove (end) other users' video sessions. Non-public Flash Streaming Servers (SFSs) can be configured and they will not be used by the existing Block / Unblock Camera to Public command; however, a user will be able to block or unblock a camera's stream within any SFS (public or non-public) on an individual basis.
- G4 RTMS: R9 provides support for a new model of TSS, the G4 RTMS by Image Sensing Systems (ISS) Canada. The G4 RTMS is a newer version of the existing TSS model, known until now simply as an RTMS, but to be known henceforth as the X3 RTMS. Additionally, R9 provides support for multi-drop deployment of TSSs, that is, two TSSs sharing the same IP address and TCP port, or sharing the same telephone number. Starting in R9, CHART will serialize communication between multi-drop TSSs, so that polling one does not collide with polling the other.

## 1.2 Objectives

The main objective of this detailed design document is to provide software developers with a framework in which to implement the requirements identified in the CHART R9 document. A matrix mapping requirements to the design is presented in Section 11 (Mapping to Requirements).

## 1.3 Scope

This design is limited to Release 9 of the CHART. It addresses both the design of the server components of CHART and the Graphical User Interface (GUI) components of CHART to support the new features being added. This design does not include designs for components implemented in earlier releases of the CHART system.

## 1.4 Design Process

The design was created by capturing the requirements of the system in UML Use Case diagrams. Class diagrams were generated showing the high level objects that address the Use Cases. Sequence diagrams were generated to show how each piece of major functionality will be achieved. This process was iterative in nature – the creation of sequence diagrams sometimes caused re-engineering of the class diagrams, and vice versa.

## 1.5 Design Tools

The work products contained within this design will be extracted from the Tau Unified Modeling Language (UML) Suite design tool. Within this tool, the design is located in the CHART project, Release 9, Analysis phase and System Design phase.

## 1.6 Work Products

The final CHART Release 9 design consists of the following work products:

- Use Case diagrams that capture the requirements of the system
- Human-Machine Interface section which provides descriptions of the screens that are changing or being added in order to allow the user to perform the described uses.
- UML Class diagrams, showing the software objects which allow the system to accommodate the uses of the system described in the Use Case diagrams
- UML Sequence diagrams showing how the classes interact to accomplish major functions of the system
- Requirement Verification Traceability Matrix that shows how this design meets the documented requirements for this feature

This document incorporates the three features by providing a Use Cases and Detailed Design section for each feature. For instance, for Decision Support, Use Cases are in Section 4, and Detailed Design (including Human-Machine Interface, Class Diagrams, and Sequence Diagrams) are in Section 5. Sections 6 & 7 cover Desktop Video, and so on.

## 2 Architecture

---

The sections below discuss specific elements of the architecture and software components that are created, changed, or used in CHART Release 9.

### 2.1 Network/Hardware

CHART Release 9 features do not impact the network or hardware architecture of the CHART system.

### 2.2 Software

CHART uses the Common Object Request Broker Architecture (CORBA) as the base architecture, with custom built software objects made available on the network allowing their data to be accessed via well defined CORBA interfaces. Communications to remote devices use the Field Management Server (FMS) architecture. Newer external interfaces such as the User Management web service, Data Exporter, and GIS service employ a web services architecture combining an HTTP request/response structure to pass XML messages.

Except where noted in the subsections below, CHART Release 9 features do not impact the software architecture of the CHART System.

#### 2.2.1 COTS Products

##### 2.2.1.1 CHART

CHART uses numerous COTS products for both run-time and development. There are no new products being added in Release 9.

The following table contains existing COTS products that have not changed for CHART Release 9:

Product Name	Description
Apache ActiveMQ	CHART uses this to connect to RITIS JMS queues
Apache Jakarta Ant	CHART uses Apache Jakarta Ant 1.6.5 to build CHART applications and deployment jars.
Apache Tomcat	CHART uses Apache Tomcat 6.0.29 as the GUI web server.
Apache XML-RPC	CHART uses the apache xmlrpc java library 3.1.2 protocol that uses XML over HTTP to implement remote procedure calls. The video Flash streaming “red button” (“kill switch”) API uses XML over HTTP remote procedure calls.
Attention! CC	CHART uses Attention! CC Version 2.1 to provide notification services.
Attention! CC API	CHART uses Attention! CC API Version 2.1 to interface

Product Name	Description
	with Attention! CC.
Attention! NS	CHART uses Attention! NS Version 7.0 to provide notification services.
Bison/Flex	CHART uses Bison and Flex as part of the process of compiling binary macro files used for performing camera menu operations on Vicon Surveyor VFT cameras.
bsn.autosuggest	The EORS integration feature uses version 2.1.3 of the bsn.autosuggest JavaScript code from brandspankingnew.net. This tool is freely available and is included as source code in the CHART GUI. It provides a simple JavaScript tool that can be associated with a text entry field. When the user types characters in the field, the tool waits until there has been no typing for a configurable number of milliseconds (to make sure the user is done typing) then places an AJAX call to a web server which can return suggested results that match the user entered text. The bsn.autosuggest tool then parses the results (XML or JSON) and displays a UI element that shows the user the suggestions and lets them select one of them by clicking on it. If a suggested element is selected by the user, a configurable JS method is invoked to allow the application to use the selected suggestion.
CoreTec Decoder Control	CHART uses a CoreTec supplied decoder control API for commanding CoreTec decoders.
Dialogic API	CHART uses the Dialogic API for sending and receiving Dual Tone Multi Frequency (DTMF) tones for HAR communications.
Flex3 SDK	The CHART GUI will use the Flex3 SDK, version 3.3 to provide the Flex compiler, the standard Flex libraries, and examples for building Flex applications.
GIF89 Encoder	Utility classes that can create .gif files with optional animation. This utility is used for the creation of DMS True Display windows.
JAXB	CHART uses the jaxb java library to automate the tedious task of hand-coding field-by-field XML translation and validation for exported data.
JDOM	CHART uses JDOM b7 (beta-7) dated 2001-07-07. JDOM provides a way to represent an XML document for easy and efficient reading, manipulation, and writing.
JacORB	CHART uses a compiled, patched version of JacORB 2.3.1. The JacORB source code, including the patched code, is kept in the CHART source repository.
Java Run-Time (JRE)	CHART uses 1.6.0_21

Product Name	Description
JavaService	CHART uses JavaService to install the server side Java software components as Windows services.
JAXEN	CHART uses JAXEN 1.0-beta-8 dated 2002-01-09. The Jaxen project is a Java XPath Engine. Jaxen is a universal object model walker, capable of evaluating XPath expressions across multiple models.
JoeSNMP	CHART uses JoeSNMP version 0.2.6 dated 2001-11-11. JoeSNMP is a Java based implementation of the SNMP protocol. CHART uses for commanding iMPath MPEG-2 decoders and for communications with NTCIP DMSs.
JSON-simple	CHART uses the JSON-simple java library to encode/decode strings that use JSON (JavaScript Object Notation).
JTS	CHART uses the Java Topology Suite (JTS) version 1.8.0 for geographical utility classes.
Log4J	CHART uses the log4J version 1.2.15 for logging purposes.
NSIS	CHART uses the Nullsoft Scriptable Installation System (NSIS), version 2.45, as the server side installation package.
Nuance Text To Speech	For text-to-speech (TTS) conversion CHART uses a TTS engine that integrates with Microsoft Speech Application Programming Interface (MSSAPI), version 5.1. CHART uses Nuance Vocalizer 4.0 with Nuance SAPI 5.1 Integration for Nuance Vocalizer 4.0.
OpenLayers	The Integrated Map feature uses the Open Layers JavaScript API 2.10 ( <a href="http://openlayers.org/">http://openlayers.org/</a> ) in order to render interactive maps within a web application without relying on vendor specific software. Open Layers is an open source product released under a BSD style license which can be found at ( <a href="http://svn.openlayers.org/trunk/openlayers/license.txt">http://svn.openlayers.org/trunk/openlayers/license.txt</a> ).
Oracle	CHART uses Oracle 10.1.0.5 as its database and uses the Oracle 10G JDBC libraries (ojdbc1.4.jar) for all database transactions.
O'Reilly Servlet	Provides classes that allow the CHART GUI to handle file uploads via multi-part form submission.
Prototype Javascript Library	The CHART GUI uses the Prototype JavaScript library, version 1.6.1, a cross-browser compatible JavaScript library provides many features (including easy Ajax support).



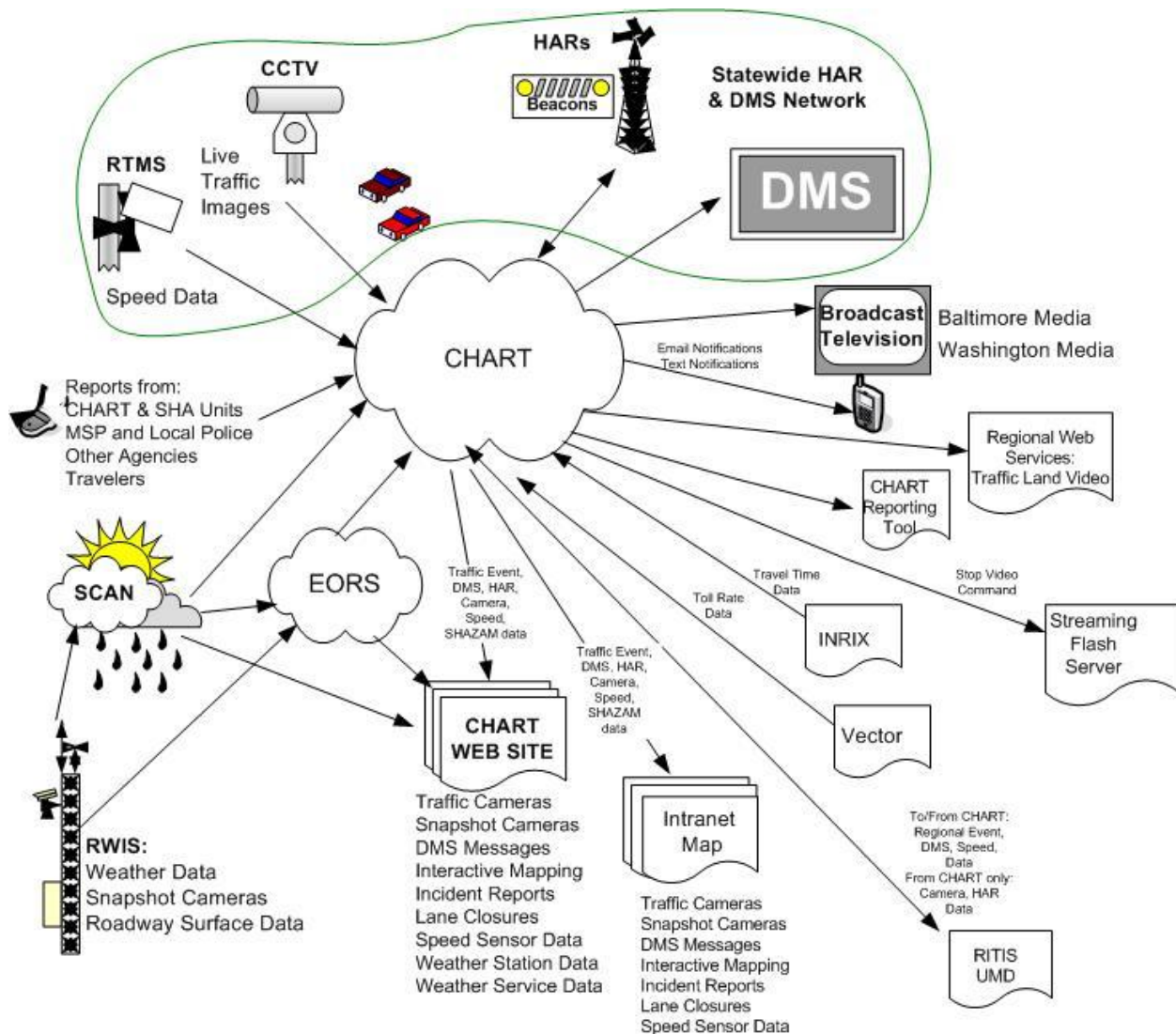
Product Name	Description
SAXPath	CHART uses SAXPath 1.0-beta-6 dated 2001-09-27. SAXPath is an event-based API for XPath parsers, that is, for parsers which parse XPath expressions.
SQLServer JDBC Driver	CHART uses this driver to lookup GIS related data and also to store Location Aliases in SQL Server databases.
Velocity Template Engine	Provides classes that CHART GUI uses in order to create dynamic web pages using velocity templates, CHART uses Velocity version 1.6.1 and tools version 1.4.
Vicon V1500 API	CHART uses a Vicon supplied API for commanding the ViconV1500 CPU to switch video on the Vicon V1500 switch

## 2.2.2 Deployment /Interface Compatibility

### 2.2.2.1 CHART

#### 2.2.2.1.1 External Interfaces

This section describes the external interfaces being added in Release 9 of the CHART.

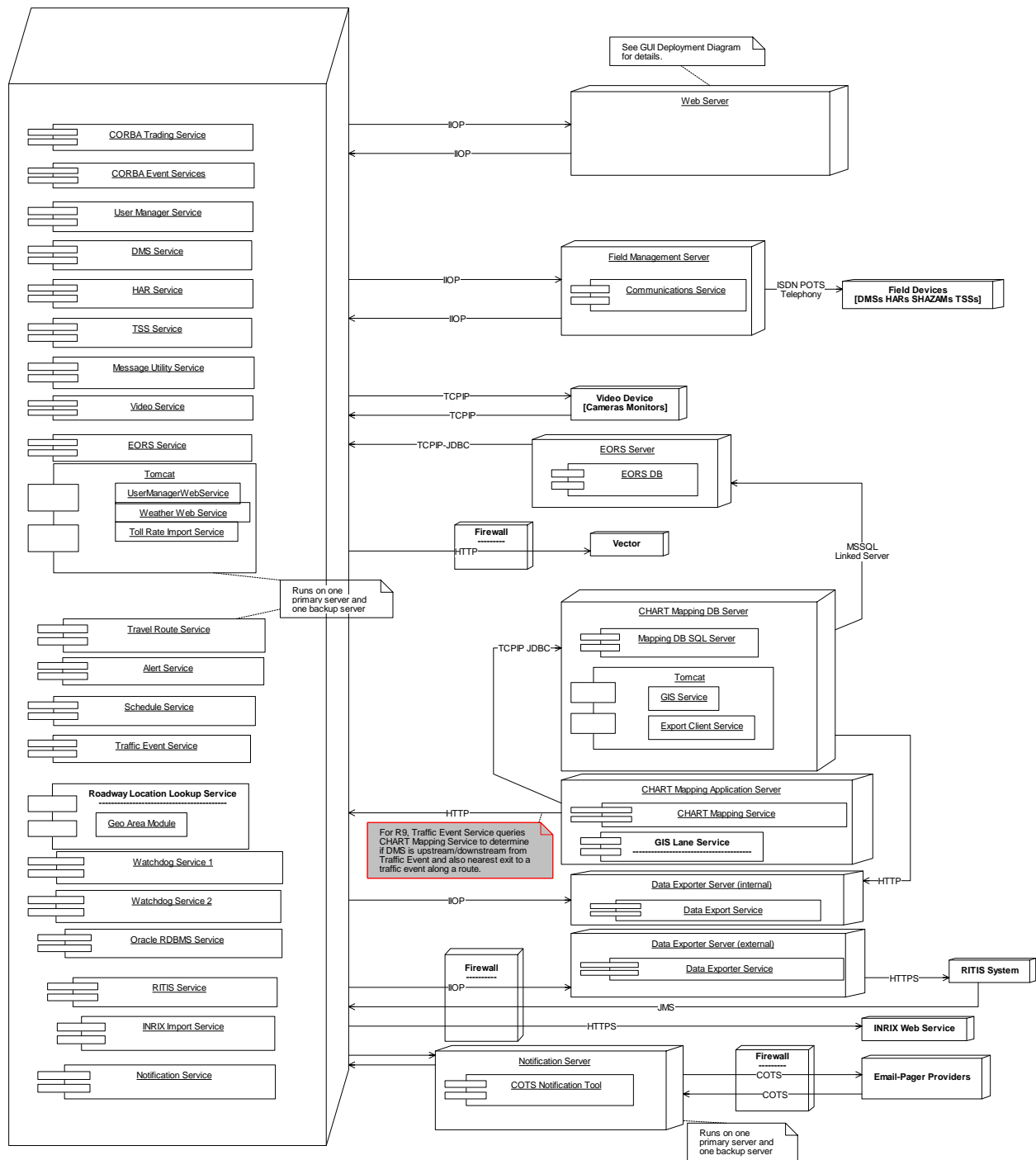


**Figure 2-1 CHART and External Interfaces**

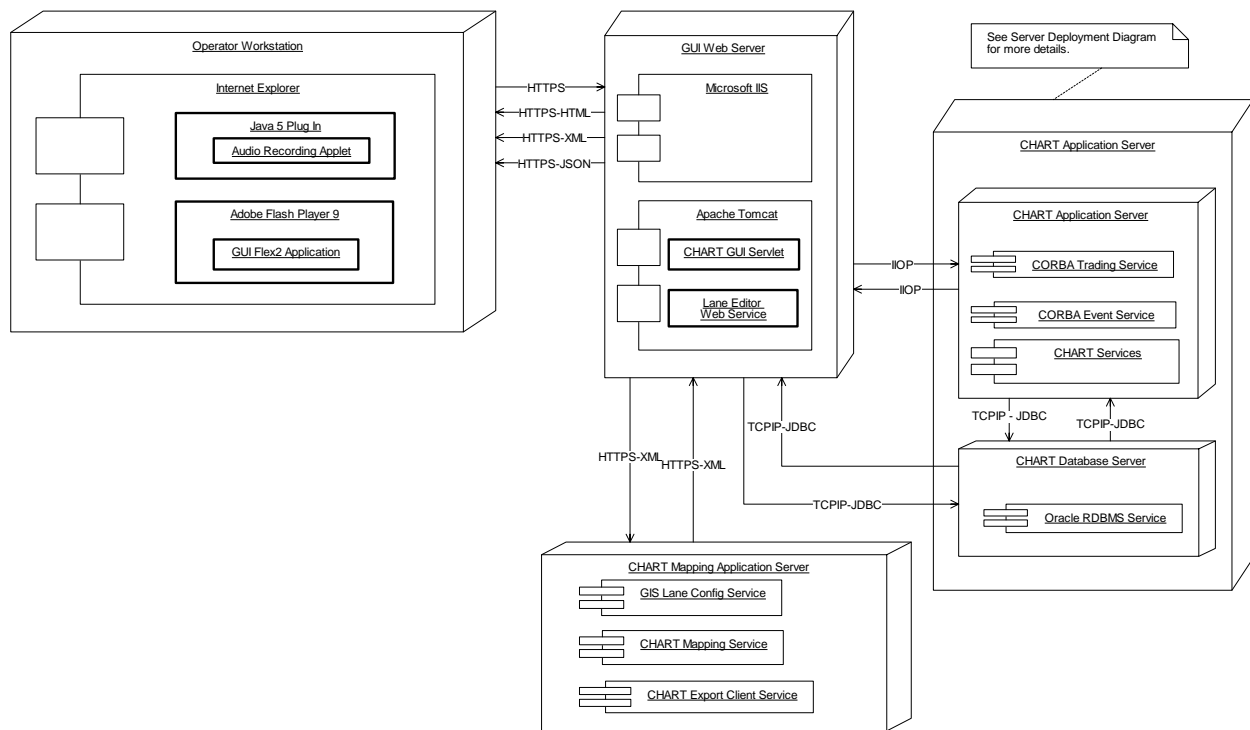
The external interfaces modified/added for R9 are:

1. The R9 Desktop Video feature changes the Streaming Flash Server configuration settings for a camera. The CHART Data Exporter will be updated to export the SFS settings per camera, as described in the CHART Data Exporter Interface Control Document (ICD).
2. For R9, the RTMS icon shown at the top of the diagram can be the original X3 RTMS or the new G4 RTMS. (Specific models of devices are not shown on this diagram.)

Server and GUI deployment diagrams are shown in the next two figures. There are no changes to the GUI deployment for R9.



**Figure 2-2 R9 Server Deployment**



**Figure 2-3 R9 GUI Deployment**

#### 2.2.2.1.2 Internal Interfaces

This section describes the internal interfaces being added or modified in Release 9 of the CHART system.

1. The R9 Decision Support feature utilizes the existing GUI interface. The CHART system IDL has been altered to support the new DecisionSupport functionality:
  - Decision Support IDL added
  - Message Template IDL: existing interfaces (used for DMS travel times and toll rate templates) modified to support Decision Support templates
  - Traffic Event Management IDL: minor changes to add decision support functionality
  - Common IDL: added proximity definitions
2. The R9 Desktop Video feature enhances the GUI to support the displaying of video in users' browser windows. The CHART system IDL has been altered to support this functionality:
  - Resource Management IDL: changes to support the management of video sessions
  - Camera Control IDL: changes to support non-public Streaming Flash Servers (SFSs) and added the ability to block / unblock per SFS

3. The existing GUI interface is changed for the TSS features. The forms that are used to configure these devices and display details about these devices are changed. These changes are detailed in the Human Machine Interface section of this document. The CHART system IDL has been altered to allow the GUI to pass the new configuration information for the TSSs to the TSS Service for persistence.

## **2.3 Security**

This section describes the security being added or modified in Release 9 of the CHART system. Unless otherwise noted, features being added for CHART Release 9 do not change security aspects of the CHART system.

## **2.4 Data**

CHART Release 9 will be tested with the currently fielded Oracle database patches.

### **2.4.1 Data Storage**

The CHART System stores most of its data in an Oracle database. Additionally the Integrated Map feature adds the ability to store location aliases to the spatial SQL Server database. Some data is stored in flat files on the CHART servers. Up until R9 TSS traffic data has been written first to flat files, and then loaded into the archive database once per night. In R9, TSS traffic data will be written to the live CHART database, and then archived to the archive database periodically (expected to be every 4 hours).

This section describes all of these types of data.

#### **2.4.1.1 Database**

##### **2.4.1.1.1 Database Architecture**

Except as noted CHART Release 9 features do not impact the overall architecture of the CHART database. We will be re-designing the TSS data load, currently is using SQL loader to load the data, we will be adding one package to load TSS data, directly from live database sites.

##### **2.4.1.1.2 Logical Design**

###### **2.4.1.1.2.1 CHART Entity Relationship Diagram (ERD)**

CHART Database entity relationship diagrams are shown below in the multiple pages of figures labeled collectively as one Figure.







# VIEWS

## VW TOLL RAW DATA

```
TOLL DATA IMPORT ID: TOLL RAW DATA TOLL DATA IMPORT ID
EXT SYS START ID: TOLL RAW DATA EXT SYS START ID
EXT SYS END ID: TOLL RAW DATA EXT SYS END ID
EXT SYS ROUTE DESC: TOLL RAW DATA EXT SYS ROUTE DESC
TOLL RATE EFF TIME: TOLL RAW DATA TOLL RATE EFF TIME
TOLL RATE EXP TIME: TOLL RAW DATA TOLL RATE EXP TIME
TOLL RATE CENTS: TOLL RAW DATA TOLL RATE CENTS
DB CODE: TOLL RAW DATA DB CODE
ARCHIVED DATE: TOLL RAW DATA ARCHIVED DATE
GET ROWID: TOLL RAW DATA GET ROWID
HOURS BEFORE ARCHIVED LIVE: TOLL RAW DATA HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW LINK DATA IMPORT

```
IMPORT ID: LINK DATA IMPORT IMPORT ID
SYSTEM TIMESTAMP: LINK DATA IMPORT SYSTEM TIMESTAMP
EXT SYS NAME: LINK DATA IMPORT EXT SYS NAME
DB CODE: LINK DATA IMPORT DB CODE
ARCHIVED DATE: LINK DATA IMPORT ARCHIVED DATE
GET ROWID: LINK DATA IMPORT GET ROWID
HOURS BEFORE ARCHIVED LIVE: LINK DATA IMPORT HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW LINK SMOOTHED DATA

```
LINK DATA IMPORT ID: LINK SMOOTHED DATA LINK DATA IMPORT ID
EXT LINK ID: LINK SMOOTHED DATA EXT LINK ID
LINK TRAVEL TIME EFF TIME: LINK SMOOTHED DATA LINK TRAVEL TIME EFF TIME
LINK TRAVEL TIME SECS: LINK SMOOTHED DATA LINK TRAVEL TIME SECS
LINK TRAVEL TIME QUAL: LINK SMOOTHED DATA LINK TRAVEL TIME QUAL
LINK SPEED MPH: LINK SMOOTHED DATA LINK SPEED MPH
DB CODE: LINK SMOOTHED DATA DB CODE
ARCHIVED DATE: LINK SMOOTHED DATA ARCHIVED DATE
GET ROWID: LINK SMOOTHED DATA GET ROWID
HOURS BEFORE ARCHIVED LIVE: LINK SMOOTHED DATA HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW LINK TRAVEL TIME

```
RL LINK ID: LINK TRAVEL TIME RL LINK ID
LINK TRAVEL TIME EFF TIME: LINK TRAVEL TIME LINK TRAVEL TIME EFF TIME
LINK TRAVEL TIME SECS: LINK TRAVEL TIME LINK TRAVEL TIME SECS
LINK TRAVEL TIME QUAL: LINK TRAVEL TIME LINK TRAVEL TIME QUAL
LINK TRAVEL TIME TREND: LINK TRAVEL TIME LINK TRAVEL TIME TREND
DB CODE: LINK TRAVEL TIME DB CODE
ARCHIVED DATE: LINK TRAVEL TIME ARCHIVED DATE
GET ROWID: LINK TRAVEL TIME GET ROWID
HOURS BEFORE ARCHIVED LIVE: LINK TRAVEL TIME HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW LINK RAW DATA

```
LINK DATA IMPORT ID: LINK RAW DATA LINK DATA IMPORT ID
EXT LINK ID: LINK RAW DATA EXT LINK ID
LINK TRAVEL TIME EFF TIME: LINK RAW DATA LINK TRAVEL TIME EFF TIME
LINK TRAVEL TIME SECS: LINK RAW DATA LINK TRAVEL TIME SECS
LINK TRAVEL TIME QUAL: LINK RAW DATA LINK TRAVEL TIME QUAL
LINK SPEED MPH: LINK RAW DATA LINK SPEED MPH
DB CODE: LINK RAW DATA DB CODE
ARCHIVED DATE: LINK RAW DATA ARCHIVED DATE
GET ROWID: LINK RAW DATA GET ROWID
HOURS BEFORE ARCHIVED LIVE: LINK RAW DATA HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW ROUTE TOLL RATE

```
TR ROUTE ID: ROUTE TOLL RATE TR ROUTE ID
TOLL RATE EFF TIME: ROUTE TOLL RATE TOLL RATE EFF TIME
TOLL RATE EXP TIME: ROUTE TOLL RATE TOLL RATE EXP TIME
TOLL RATE CENTS: ROUTE TOLL RATE TOLL RATE CENTS
TOLL RATE REASON CODE: ROUTE TOLL RATE TOLL RATE REASON CODE
TOLL RATE INAPPLICABLE IND: ROUTE TOLL RATE TOLL RATE INAPPLICABLE IND
DB CODE: ROUTE TOLL RATE DB CODE
ARCHIVED DATE: ROUTE TOLL RATE ARCHIVED DATE
GET ROWID: ROUTE TOLL RATE GET ROWID
HOURS BEFORE ARCHIVED LIVE: ROUTE TOLL RATE HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW ROUTE TRAVEL TIME

```
TR ROUTE ID: ROUTE TRAVEL TIME TR ROUTE ID
ROUTE TRAVEL TIME EFF TIME: ROUTE TRAVEL TIME ROUTE TRAVEL TIME EFF TIME
ROUTE TRAVEL TIME SECS: ROUTE TRAVEL TIME ROUTE TRAVEL TIME SECS
ROUTE TRAVEL TIME QUAL: ROUTE TRAVEL TIME ROUTE TRAVEL TIME QUAL
ROUTE TRAVEL TIME TREND: ROUTE TRAVEL TIME ROUTE TRAVEL TIME TREND
TRAVEL TIME INAPPLICABLE IND: ROUTE TRAVEL TIME TRAVEL TIME INAPPLICABLE IND
DB CODE: ROUTE TRAVEL TIME DB CODE
ARCHIVED DATE: ROUTE TRAVEL TIME ARCHIVED DATE
GET ROWID: ROUTE TRAVEL TIME GET ROWID
HOURS BEFORE ARCHIVED LIVE: ROUTE TRAVEL TIME HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## ROUTE TRAVEL TIME

```
TR ROUTE ID: CHAR(32)
ROUTE TRAVEL TIME EFF TIME: DATE
ROUTE TRAVEL TIME SECS: NUMBER(9)
ROUTE TRAVEL TIME TREND: NUMBER(1)
TRAVEL TIME INAPPLICABLE IND: NUMBER(1)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

```
TR ROUTE ID: CHAR(32)
ROUTE TRAVEL TIME EFF TIME: DATE
ROUTE TRAVEL TIME CALC: VARCHAR2(1000)
ROUTE TRAVEL TIME REASON CODE: NUMBER(2)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

```
SYSTEM TIMESTAMP: DATE
DMS DEVICE ID: CHAR(32)
DEVICE NAME: VARCHAR2(19)
SCHEDULE CONFIG FLAG: NUMBER(2)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## VW DMS TRAV RT MSG CONFIG LOG

```
SYSTEM TIMESTAMP: DMS TRAV ROUTE MSG CONFIG LOG SYSTEM TIMESTAMP
DMS DEVICE ID: DMS TRAV ROUTE MSG CONFIG LOG DMS DEVICE ID
DEVICE NAME: DMS TRAV ROUTE MSG CONFIG LOG DEVICE NAME
SCHEDULE CONFIG FLAG: DMS TRAV ROUTE MSG CONFIG LOG SCHEDULE_CONFIG_FLAG
DB CODE: DMS TRAV ROUTE MSG CONFIG LOG DB CODE
ARCHIVED DATE: DMS TRAV ROUTE MSG CONFIG LOG ARCHIVED DATE
GET ROWID: DMS TRAV ROUTE MSG CONFIG LOG GET ROWID
HOURS BEFORE ARCHIVED LIVE: DMS TRAV ROUTE MSG CONFIG LOG HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW DMS TRAV RT MSG ROUTES LOG

```
SYSTEM TIMESTAMP: DMS TRAV ROUTE MSG ROUTES LOG SYSTEM TIMESTAMP
DMS DEVICE ID: DMS TRAV ROUTE MSG ROUTES LOG DMS DEVICE ID
DMS TRAV ROUTE MSG ID: DMS TRAV ROUTE MSG ROUTES LOG DMS TRAV ROUTE MSG ID
TRAV ROUTE ID: DMS TRAV ROUTE MSG ROUTES LOG TRAV ROUTE ID
DB CODE: DMS TRAV ROUTE MSG ROUTES LOG DB CODE
ARCHIVED DATE: DMS TRAV ROUTE MSG ROUTES LOG ARCHIVED DATE
GET ROWID: DMS TRAV ROUTE MSG ROUTES LOG GET ROWID
HOURS BEFORE ARCHIVED LIVE: DMS TRAV ROUTE MSG ROUTES LOG HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW DMS TRAV RT MSG MSGS LOG

```
SYSTEM TIMESTAMP: DMS TRAV ROUTE MSG MSGS LOG SYSTEM TIMESTAMP
MSGS LOG SEQUENCE: DMS TRAV ROUTE MSG MSGS LOG MSGS_LOG_SEQUENCE
DMS DEVICE ID: DMS TRAV ROUTE MSG MSGS LOG DMS DEVICE ID
DMS TRAV ROUTE MSG ID: DMS TRAV ROUTE MSG MSGS LOG DMS TRAV ROUTE MSG ID
DMS TRAV ROUTE MSG TEMPLATE ID: DMS TRAV ROUTE MSG MSGS LOG DMS TRAV ROUTE MSG TEMPLATE ID
DB CODE: DMS TRAV ROUTE MSG MSGS LOG DB CODE
ARCHIVED DATE: DMS TRAV ROUTE MSG MSGS LOG ARCHIVED DATE
GET ROWID: DMS TRAV ROUTE MSG MSGS LOG GET ROWID
HOURS BEFORE ARCHIVED LIVE: DMS TRAV ROUTE MSG MSGS LOG HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW DMS TRAV RT MSG STATUS LOG

```
SYSTEM TIMESTAMP: DMS TRAV ROUTE MSG STATUS LOG SYSTEM TIMESTAMP
STAT LOG SEQUENCE: DMS TRAV ROUTE MSG STATUS LOG STAT_LOG_SEQUENCE
DMS DEVICE ID: DMS TRAV ROUTE MSG STATUS LOG DMS DEVICE ID
DEVICE NAME: DMS TRAV ROUTE MSG STATUS LOG DEVICE NAME
COMMUNICATION MODE: DMS TRAV ROUTE MSG STATUS LOG COMMUNICATION MODE
OPERATIONAL STATUS: DMS TRAV ROUTE MSG STATUS LOG OPERATIONAL STATUS
SCHEDULE ENABLED INDICATOR: DMS TRAV ROUTE MSG STATUS LOG SCHEDULE_ENABLED_INDICATOR
ENABLED DMS TRAV ROUTE MSG ID: DMS TRAV ROUTE MSG STATUS LOG ENABLED_DMS_TRAV_ROUTE_MSG_ID
DMS MESSAGE: DMS TRAV ROUTE MSG STATUS LOG DMS MESSAGE
DMS TRAV ROUTE MSG STATE: DMS TRAV ROUTE MSG STATUS LOG DMS TRAV ROUTE MSG STATE
DMS TRAV ROUTE MSG REASON: DMS TRAV ROUTE MSG STATUS LOG DMS TRAV ROUTE MSG REASON
DB CODE: DMS TRAV ROUTE MSG STATUS LOG DB CODE
ARCHIVED DATE: DMS TRAV ROUTE MSG STATUS LOG ARCHIVED DATE
GET ROWID: DMS TRAV ROUTE MSG STATUS LOG GET ROWID
HOURS BEFORE ARCHIVED LIVE: DMS TRAV ROUTE MSG STATUS LOG HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## VW ROUTE TRAVEL TIME TEXT

```
TR ROUTE ID: ROUTE TRAVEL TIME TEXT TR ROUTE ID
ROUTE TRAVEL TIME EFF TIME: ROUTE TRAVEL TIME TEXT ROUTE TRAVEL TIME EFF TIME
ROUTE TRAVEL TIME CALC: ROUTE TRAVEL TIME TEXT ROUTE TRAVEL TIME CALC
ROUTE TRAVEL TIME REASON CODE: ROUTE TRAVEL TIME TEXT ROUTE TRAVEL TIME REASON_CODE
DB CODE: ROUTE TRAVEL TIME TEXT DB CODE
ARCHIVED DATE: ROUTE TRAVEL TIME TEXT ARCHIVED DATE
GET ROWID: ROUTE TRAVEL TIME TEXT GET ROWID
HOURS BEFORE ARCHIVED LIVE: ROUTE TRAVEL TIME TEXT HOURS_BEFORE_ARCHIVED_LIVE
HOURS AFTER ARCHIVED: ((Sysdate - ARCHIVED))
```

## LINK TRAVEL TIME

```
RL LINK ID: CHAR(32)
LINK TRAVEL TIME EFF TIME: DATE
LINK TRAVEL TIME SECS: NUMBER(9)
LINK TRAVEL TIME QUAL: NUMBER(2)
LINK TRAVEL TIME TREND: NUMBER(1)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## IMPORT ID: NUMBER

```
SYSTEM TIMESTAMP: DATE
EXT SYS NAME: VARCHAR2(35)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## SYSTEM TIMESTAMP: DATE

```
DMS DEVICE ID: CHAR(32)
DMS TRAV ROUTE MSG ID: CHAR(32)
TRAV ROUTE ID: CHAR(32)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## DMS TRAV ROUTE MSG MSGS LOG

```
SYSTEM TIMESTAMP: DATE
MSGS LOG SEQUENCE: NUMBER
DMS DEVICE ID: VARCHAR2(32)
DMS TRAV ROUTE MSG ID: VARCHAR2(32)
DMS TRAV ROUTE MSG TEMPLATE ID: VARCHAR2(4000)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## LINK SMOOTHED DATA

```
LINK DATA IMPORT ID: NUMBER
EXT LINK ID: CHAR(9)
LINK TRAVEL TIME EFF TIME: DATE
LINK TRAVEL TIME SECS: NUMBER(5)
LINK TRAVEL TIME QUAL: NUMBER(2)
LINK SPEED MPH: NUMBER(3)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## LINK RAW DATA

```
LINK DATA IMPORT ID: NUMBER
EXT LINK ID: CHAR(9)
LINK TRAVEL TIME EFF TIME: DATE
LINK TRAVEL TIME SECS: NUMBER(5)
LINK TRAVEL TIME QUAL: NUMBER(2)
LINK SPEED MPH: NUMBER(3)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```

## ROUTE TOLL RATE

```
TR ROUTE ID: CHAR(32)
TOLL RATE EFF TIME: DATE
TOLL RATE EXP TIME: DATE
TOLL RATE CENTS: NUMBER(9)
TOLL RATE REASON CODE: NUMBER(2)
TOLL RATE INAPPLICABLE IND: NUMBER(1)
DB CODE: CHAR(1)
ARCHIVED DATE: DATE
HOURS BEFORE ARCHIVED LIVE: NUMBER
GET ROWID: ROWID
```



**OBJECT LOCATION**

OBJECT ID: VARCHAR(32)  
LOCATION TEXT: VARCHAR(1024)  
LOCATION DESC: OVERRIDDEN: NUMBER(1)  
COUNTY NAME: VARCHAR(50)  
COUNTY FIPS CODE: CHAR(3)  
COUNTY CODE: NUMBER(3)  
USPS STATE CODE: CHAR(2)  
STATE FULL NAME: VARCHAR(32)  
STATE FIPS CODE: CHAR(2)  
REGION NAME: VARCHAR(32)  
ROUTE SPEC TYPE: NUMBER(1)  
ROUTE FREE FORM TEXT: VARCHAR(40)  
ROUTE TYPE: NUMBER(1)  
ROUTE PREFIX: VARCHAR(10)  
ROUTE NUMBER: VARCHAR(10)  
ROUTE SUFFIX: VARCHAR(10)  
INT FEAT TYPE: NUMBER(1)  
INT FEAT PROX TYPE: NUMBER(2)  
ROAD NAME: VARCHAR(50)  
INT ROUTE SPEC TYPE: NUMBER(1)  
INT ROUTE FREE FORM TEXT: VARCHAR(40)  
INT ROUTE TYPE: NUMBER(1)  
INT ROAD NAME: VARCHAR(50)  
INT ROUTE PREFIX: VARCHAR(10)  
INT ROUTE NUMBER: VARCHAR(10)  
INT ROUTE SUFFIX: VARCHAR(10)  
INT FEAT MILEPOST TYPE: NUMBER(1)  
INT FEAT MILEPOST DATA: NUMBER(6)  
ROADWAY LOC ALIAS PUB NAME: VARCHAR(90)  
ROADWAY LOC ALIAS INT NAME: VARCHAR(90)  
LATITUDE UDEG: NUMBER(10)  
LONGITUDE UDEG: NUMBER(10)  
GEOLOC SOURCE TYPE: NUMBER(2)  
GEOLOC SOURCE DESC: VARCHAR(35)  
SHOW ROUTE NAME: NUMBER(1)  
DIRECTION CODE: NUMBER(3)  
OBJECT TYPE: NUMBER(3)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
INT FEAT EXIT NUMBER: NUMBER(3)  
INT FEAT EXIT SUFFIX: VARCHAR(16)  
NT FEAT EXIT ROAD NAME: VARCHAR(64)  
SEC INT FEAT TYPE: NUMBER(1)  
SEC INT ROUTE SPEC TYPE: NUMBER(1)  
SEC INT ROUTE FREE FORM TEXT: VARCHAR(250)  
SEC INT ROUTE TYPE: NUMBER(1)  
SEC INT ROAD NAME: VARCHAR(50)  
SEC INT ROUTE PREFIX: VARCHAR(10)  
SEC INT ROUTE NUMBER: VARCHAR(10)  
SEC INT ROUTE SUFFIX: VARCHAR(10)  
SEC INT FEAT MILEPOST TYPE: NUMBER(1)  
SEC INT FEAT MILEPOST DATA: NUMBER(6)  
SEC INT FEAT EXIT NUMBER: NUMBER(3)  
SEC INT FEAT EXIT SUFFIX: VARCHAR(16)  
SEC INT FEAT EXIT ROAD NAME: VARCHAR(64)  
SHOW\_SEC\_INT\_ROUTE\_NAME: NUMBER(1)

SYSTEM TIMESTAMP: DATE  
MSG LOG SEQUENCE: NUMBER  
DMS DEVICE ID: VARCHAR(32)  
COMMUNICATION MODE: NUMBER  
OPERATIONAL STATUS: NUMBER  
SCHEDULE ENABLED INDICATOR: NUMBER  
ENABLED DMS TRAV ROUTE MSG ID: VARCHAR(32)  
DMS MESSAGE: VARCHAR(1024)  
DMS TRAV ROUTE MSG STATE: NUMBER  
DMS TRAV ROUTE MSG REASON: VARCHAR(4000)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
STAT LOG SEQUENCE: NUMBER  
GET ROWID: ROWID

SYSTEM TIMESTAMP: DATE  
MSG LOG SEQUENCE: NUMBER  
DMS DEVICE ID: VARCHAR(32)  
DMS TRAV ROUTE MSG ID: VARCHAR(32)  
DMS TRAV ROUTE MSG TEMPLATE ID: VARCHAR(24)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**LINK TRAVEL TIME**

RL LINK ID: CHAR(32)  
LINK TRAVEL TIME EFF TIME: DATE  
LINK TRAVEL TIME SECS: NUMBER(5)  
LINK TRAVEL TIME QUAL: NUMBER(3)  
LINK TRAVEL TIME TEND: NUMBER(1)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**ARCHIVE PURGED HISTORY**

DATABASE NAME: VARCHAR(10)  
PURGED DATE: DATE  
ARCH SYS PROFILE RECORD: VARCHAR(250)  
REMARKS: VARCHAR(100)

**TRAVEL ROUTE**

ROUTE ID: CHAR(32)  
NAME: VARCHAR(250)  
MILLI MILES: NUMBER(6)  
USER LOCATION IND: NUMBER(1)  
PRIMARY DEST TEXT: VARCHAR(30)  
TRAVEL TIME ENABLED IND: NUMBER(1)  
MIN TRAVEL TIME MINS: NUMBER(3)  
MAX TRAVEL TIME MINS: NUMBER(3)  
MAX BAD LINKS: NUMBER(3)  
ALERT TRAVEL TIME MINS: NUMBER(3)  
TRAV TIME ALERTS ENABLED IND: NUMBER(1)  
TRAV TIME ALERT OP CENTER: CHAR(32)  
TRAV TIME NOTIFS ENABLED IND: NUMBER(1)  
TOLL RATE ENABLED IND: NUMBER(1)  
TOLL RATE EXT SYS NAME: VARCHAR(35)  
TOLL RATE EXT START ID: VARCHAR(35)  
TOLL RATE EXT END ID: VARCHAR(35)  
TOLL RATE EXT DESC: VARCHAR(127)  
TOLL RATE ALERTS ENABLED IND: NUMBER(1)  
TOLL RATE ALERT OP CENTER: CHAR(32)  
ROUTE NOTIFS ENABLED IND: NUMBER(1)  
TOLL RATE NOTIFS RECIPIENT: VARCHAR(35)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**ROADWAY LINK**

LINK ID: CHAR(32)  
EXT SYS NAME: VARCHAR(10)  
EXT LINK ID: VARCHAR(32)  
LINK NAME: VARCHAR(100)  
USPS STATE CODE: CHAR(2)  
STATE FIPS CODE: CHAR(2)  
COUNTY NAME: VARCHAR(50)  
COUNTY FIPS CODE: VARCHAR(3)  
ROUTE SPEC TYPE: NUMBER(1)  
ROUTE FREE FORM TEXT: VARCHAR(250)  
ROUTE TYPE: NUMBER(1)  
ROUTE PREFIX: VARCHAR(10)  
ROUTE NUMBER: VARCHAR(100)  
ROUTE SUFFIX: VARCHAR(10)  
MILLI MILES: NUMBER(6)  
START LAT UDEG: NUMBER(9)  
START LONG UDEG: NUMBER(10)  
END LAT UDEG: NUMBER(9)  
END LONG UDEG: NUMBER(10)  
ROAD NAME: VARCHAR(100)  
DIRECTION CODE: NUMBER  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**TRAVEL ROUTE LOCATION**

TR ROUTE ID: VARCHAR(32)  
SORT ORDER NUMBER: NUMBER(2)  
COUNTY NAME: VARCHAR(50)  
COUNTY FIPS CODE: CHAR(3)  
COUNTY CODE: NUMBER(3)  
DIRECTION CODE: NUMBER(3)  
USPS STATE CODE: CHAR(2)  
STATE FULL NAME: VARCHAR(32)  
STATE FIPS CODE: CHAR(2)  
ROUTE SPEC TYPE: NUMBER(1)  
ROUTE FREE FORM TEXT: VARCHAR(240)  
ROUTE TYPE: NUMBER(1)  
ROUTE PREFIX: VARCHAR(10)  
ROUTE NUMBER: VARCHAR(10)  
ROUTE SUFFIX: VARCHAR(10)  
ROAD NAME: VARCHAR(50)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**DMS TRAV ROUTE MSG ROUTES LOG**

SYSTEM TIMESTAMP: DATE  
DMS DEVICE ID: CHAR(32)  
DMS TRAV ROUTE MSG ID: CHAR(32)  
TRAV ROUTE ID: CHAR(32)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**LINK RAW DATA**

LINK DATA IMPORT ID: NUMBER  
EXT LINK ID: CHAR(9)  
LINK TRAVEL TIME EFF TIME: DATE  
LINK TRAVEL TIME SECS: NUMBER(5)  
LINK TRAVEL TIME QUAL: NUMBER(2)  
LINK SPEED MPH: NUMBER(3)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**IS TRAV ROUTE MSG CONFIG LOG**

SYSTEM TIMESTAMP: DATE  
DMS DEVICE ID: CHAR(32)  
DEVICE NAME: VARCHAR(15)  
SCHEDULE CONFIG FLAG: NUMBER(2)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**LINK SMOOTHED DATA**

LINK DATA IMPORT ID: NUMBER  
EXT LINK ID: CHAR(9)  
LINK TRAVEL TIME EFF TIME: DATE  
LINK TRAVEL TIME SECS: NUMBER(5)  
LINK TRAVEL TIME QUAL: NUMBER(2)  
LINK SPEED MPH: NUMBER(3)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**LINK DATA IMPORT**

IMPORT ID: NUMBER  
SYSTEM TIMESTAMP: DATE  
EXT SYS NAME: VARCHAR(35)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE  
HOURS BEFORE ARCHIVED LIVE: NUMBER  
GET ROWID: ROWID

**TRAVEL ROUTE LINK**

TR ROUTE ID: CHAR(32)  
SORT ORDER NUMBER: NUMBER(4)  
RL LINK ID: CHAR(32)  
PERCENT: NUMBER(2)  
MIN ALLOWED QUALITY: NUMBER(3)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**ROADWAY LOCATION STATE**

NAME: VARCHAR(20)  
USPS CODE: CHAR(2)  
FIPS CODE: CHAR(2)  
DESCRIPTION: VARCHAR(1024)

**CHART ARCHIVE PARAMETERS**

DATABASE NAME: VARCHAR(10)  
KEY: VARCHAR(40)  
VALUE: NUMBER(10)  
UNITS: VARCHAR(10)  
REMARKS: VARCHAR(250)

**DMS TRAV ROUTE MSG**

DMS DEVICE ID: CHAR(32)  
MSG ID: CHAR(32)  
TEMPLATE ID: CHAR(32)  
AUTO ROW POSITIONING IND: NUMBER(1)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**EXTERNAL EVENT FILTER**

RULE ID: CHAR(32)  
RULE: VARCHAR(2048)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**TRAVEL ROUTE CONSUMER**

TR ROUTE ID: CHAR(32)  
SORT ORDER NUMBER: NUMBER(2)  
CONSUMER ID: CHAR(32)  
PROXY CONSUMER ID: CHAR(32)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**EXTERNAL OBJECT EXCLUSION**

EXCLUSION ID: CHAR(32)  
EXTERNAL OBJECT ID: VARCHAR(32)  
EXTERNAL OBJECT TYPE: NUMBER(3)  
EXTERNAL SYSTEM: VARCHAR(35)  
EXTERNAL AGENCY: VARCHAR(25)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**ROADWAY LOCATION COUNTY**

FIPS CODE: CHAR(3)  
USPS STATE CODE: CHAR(2)  
NAME: VARCHAR(50)  
CHART MAPPING CODE: CHAR(2)  
DESCRIPTION: VARCHAR(1024)

**ROADWAY LOCATION REGION**

NAME: VARCHAR(32)  
USPS STATE CODE: CHAR(2)  
DESCRIPTION: VARCHAR(1024)

**DMS TRAV TIME SCHEDULE**

DMS DEVICE ID: CHAR(32)  
START HOUR: NUMBER(2)  
START MIN: NUMBER(2)  
END HOUR: NUMBER(2)  
END MIN: NUMBER(2)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**TRAVEL ROUTE DEST**

TR ROUTE ID: CHAR(32)  
SORT ORDER NUMBER: NUMBER(1)  
ALT DEST TEXT: VARCHAR(30)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

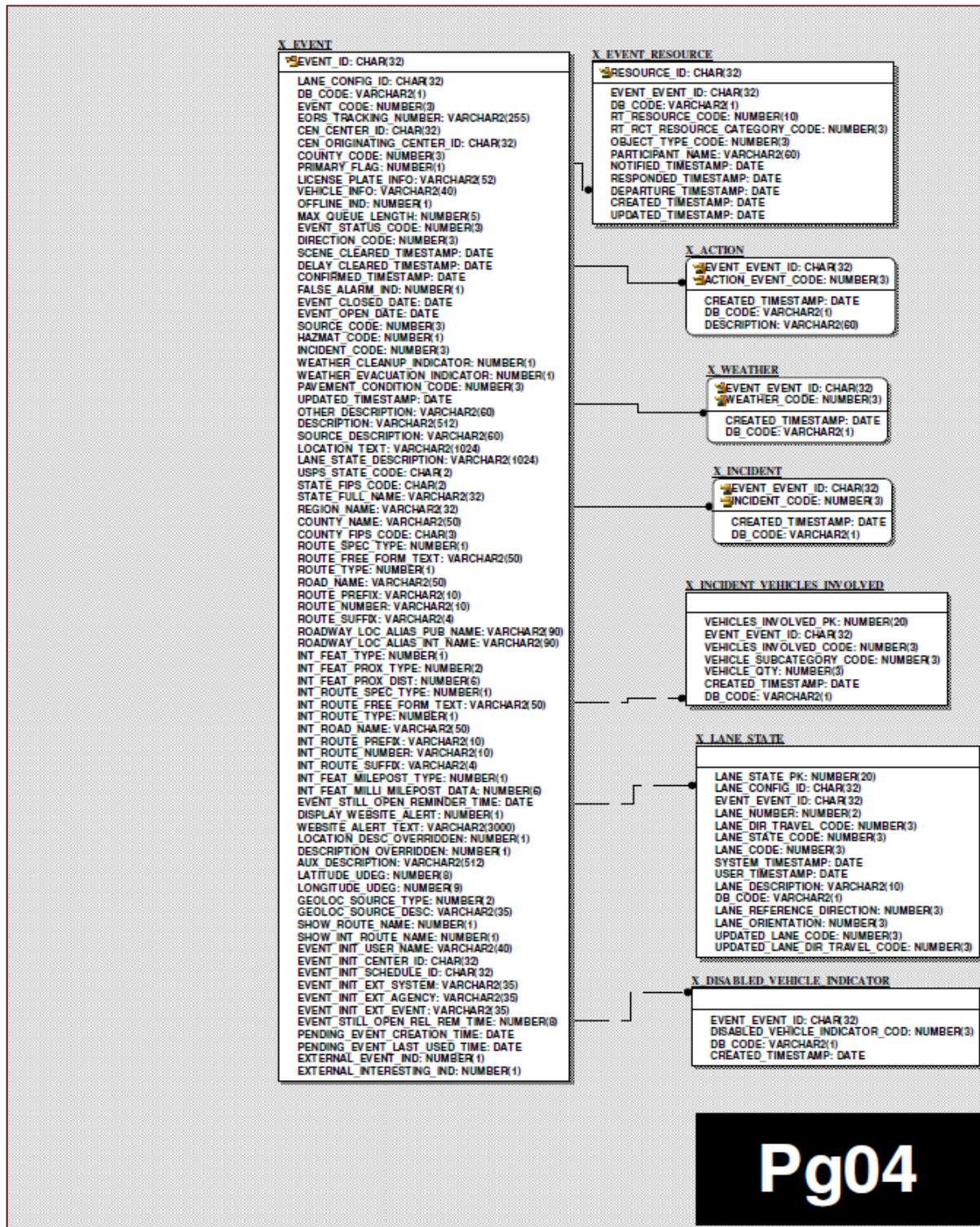
**DMS TRAV ROUTE MSG ROUTE**

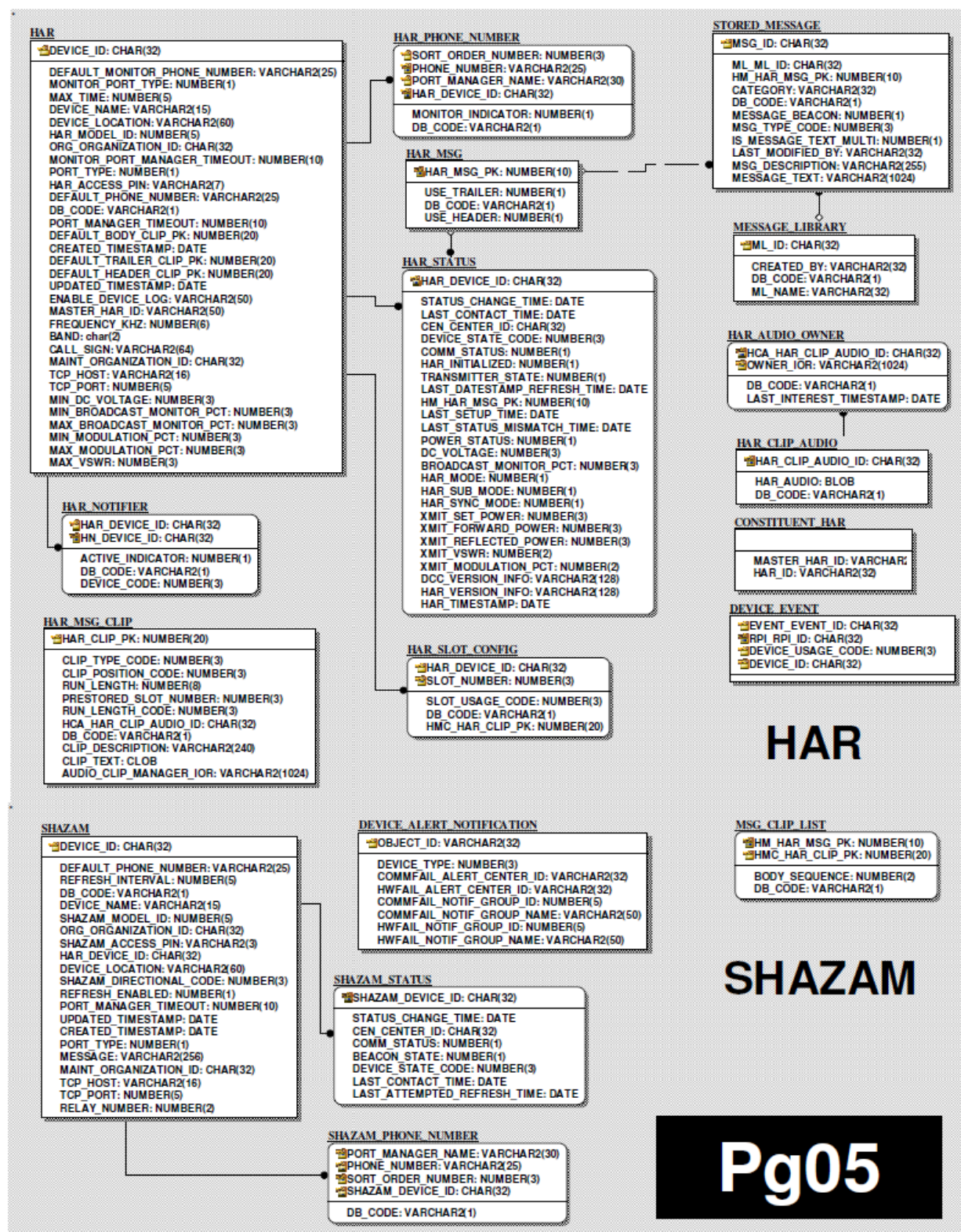
OTRM MSG ID: CHAR(32)  
TRAVEL ROUTE ID: CHAR(32)  
SORT ORDER NUM: NUMBER(2)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

**DMS RELATED ROUTE**

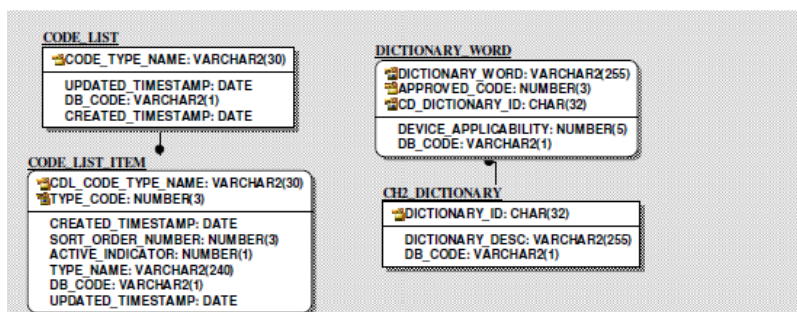
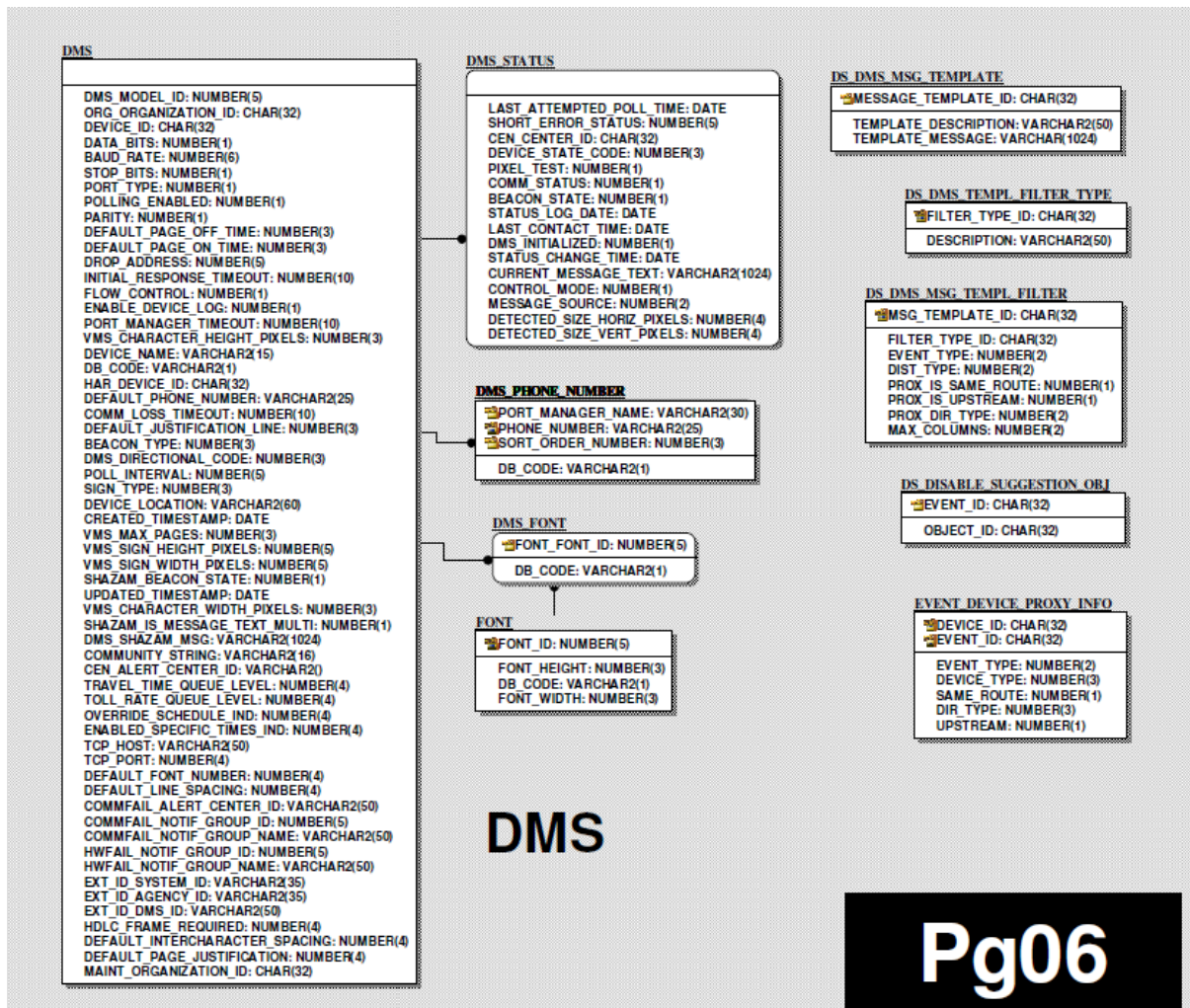
DMS DEVICE ID: CHAR(32)  
TRAVEL ROUTE ID: CHAR(32)  
DB CODE: CHAR(1)  
ARCHIVED DATE: DATE

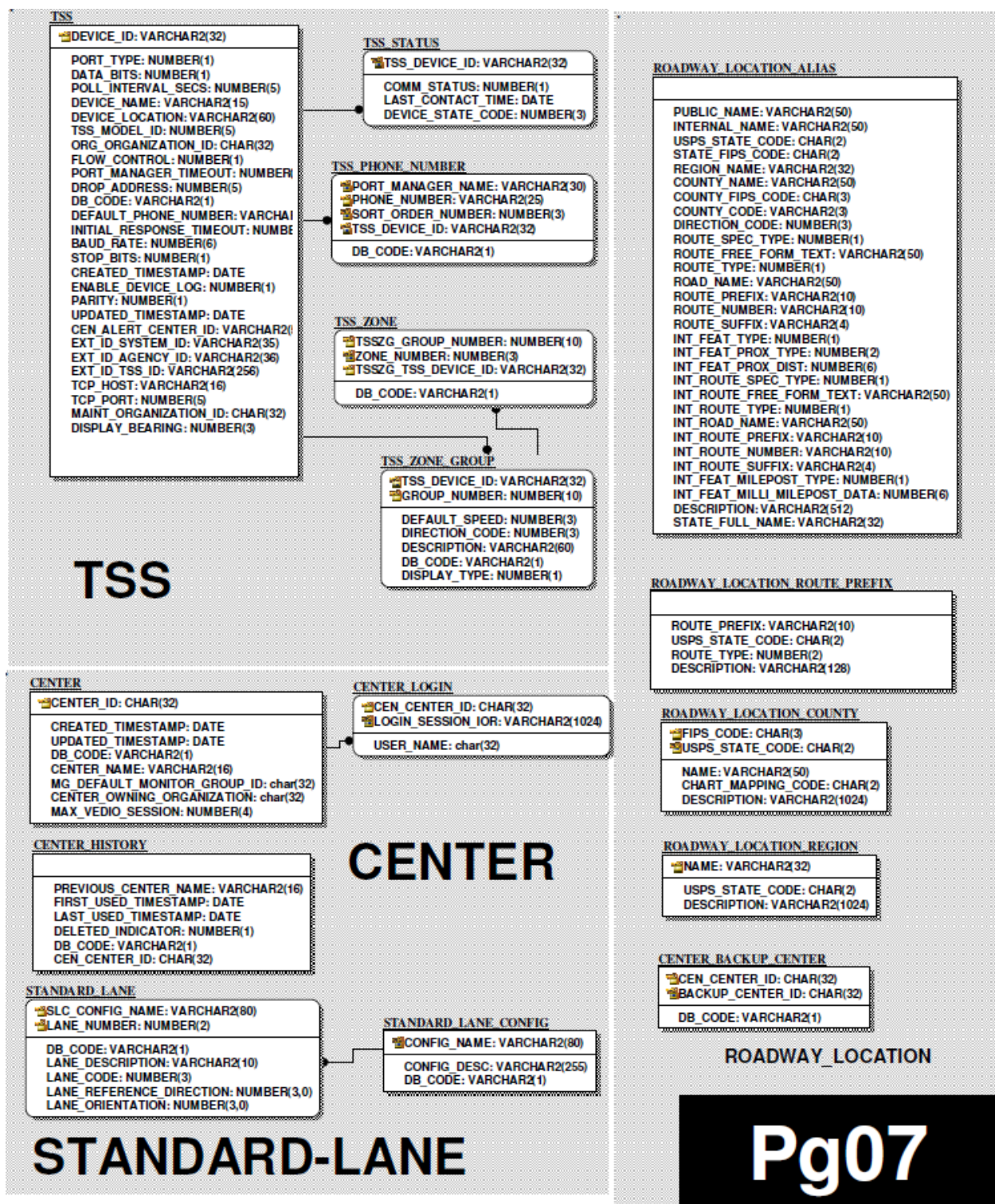




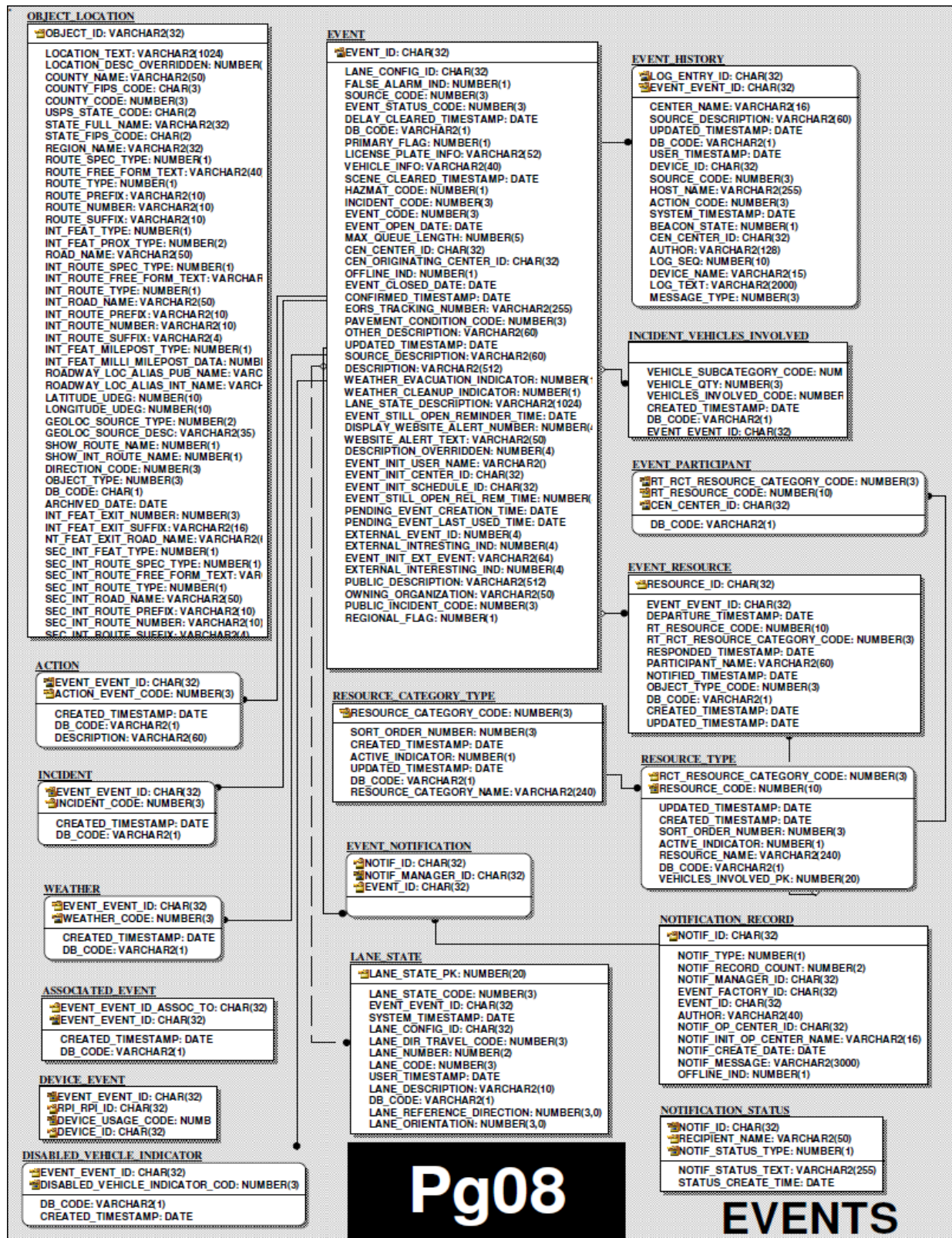








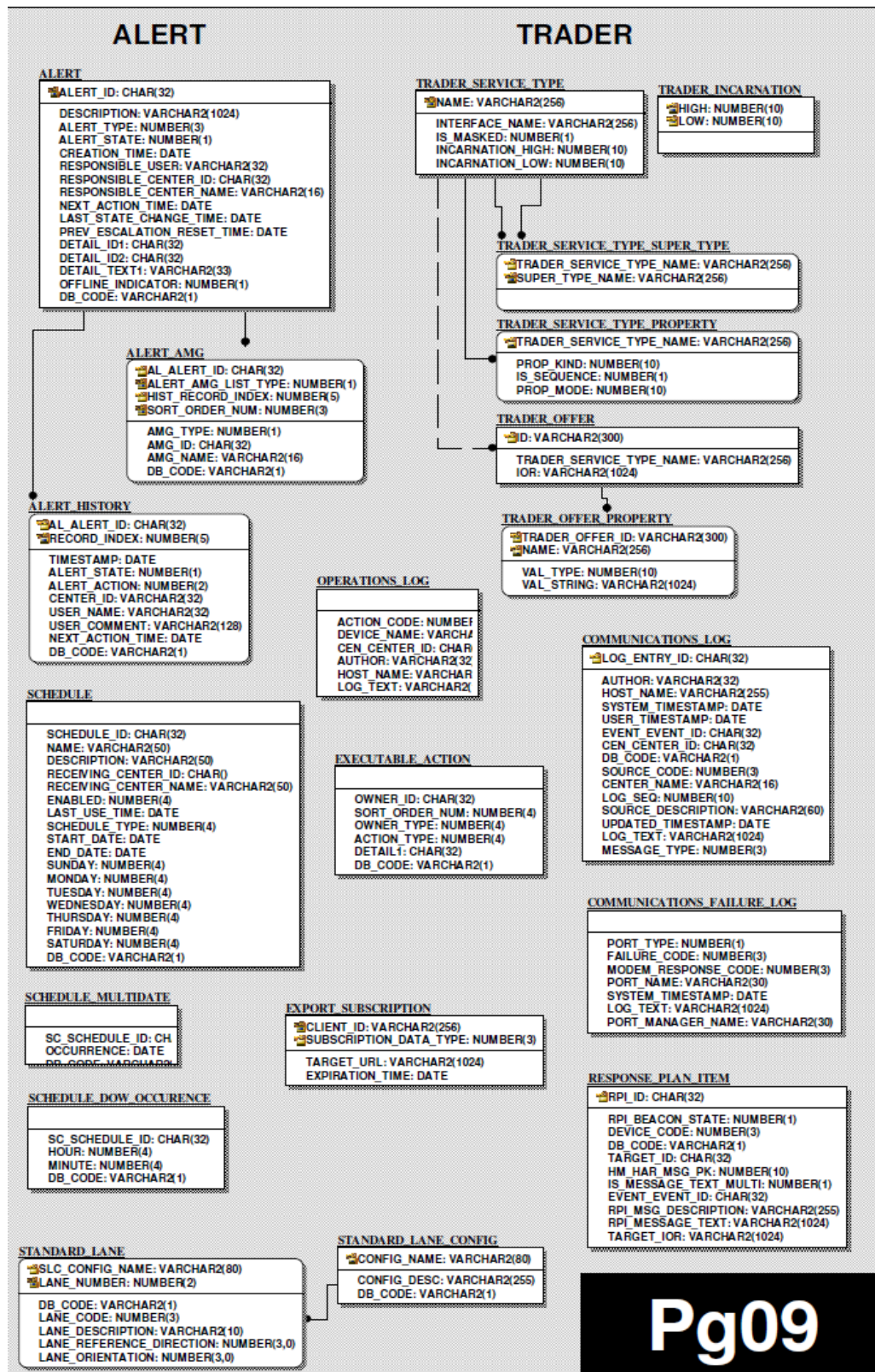




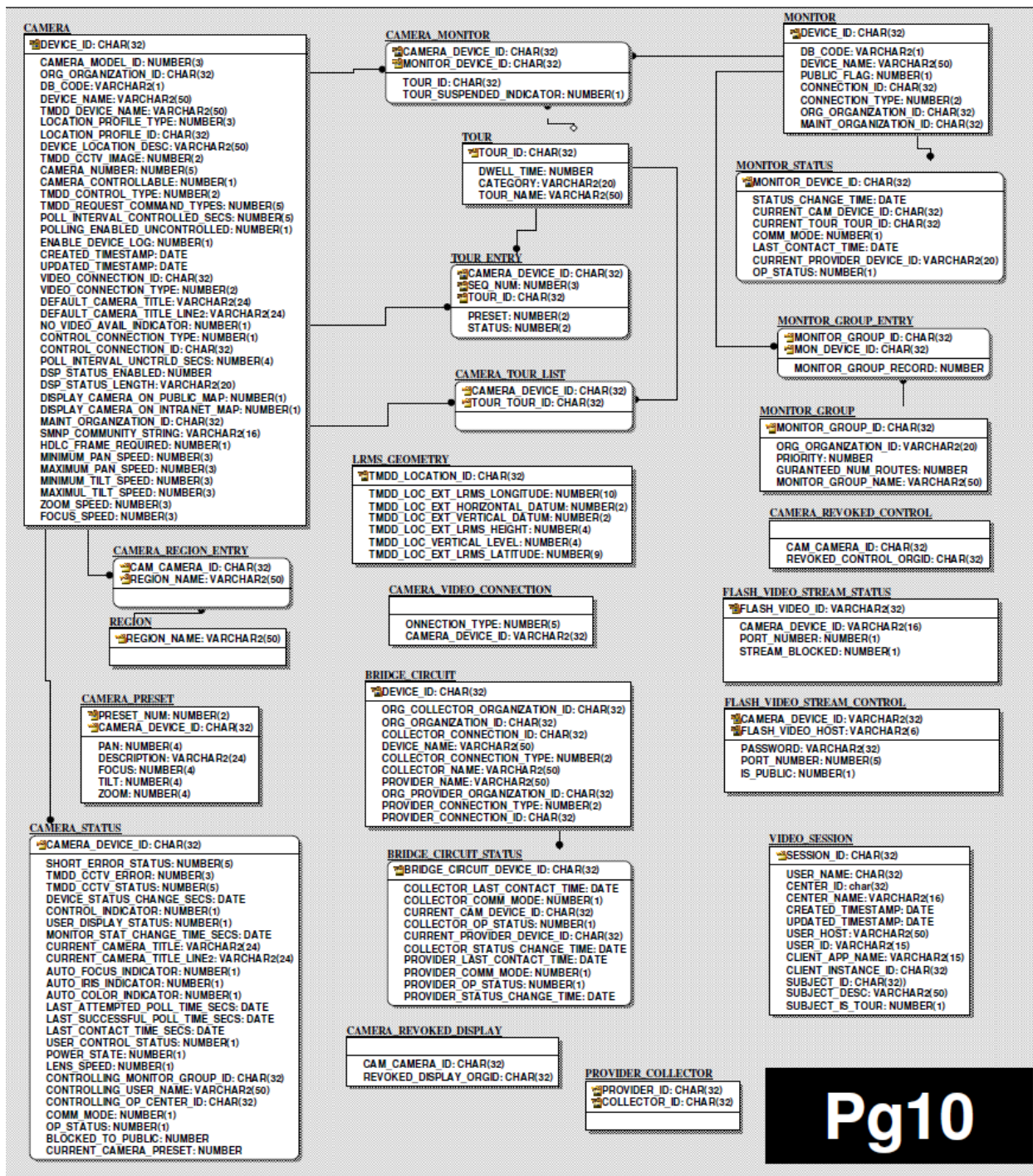
Pg08

EVENTS

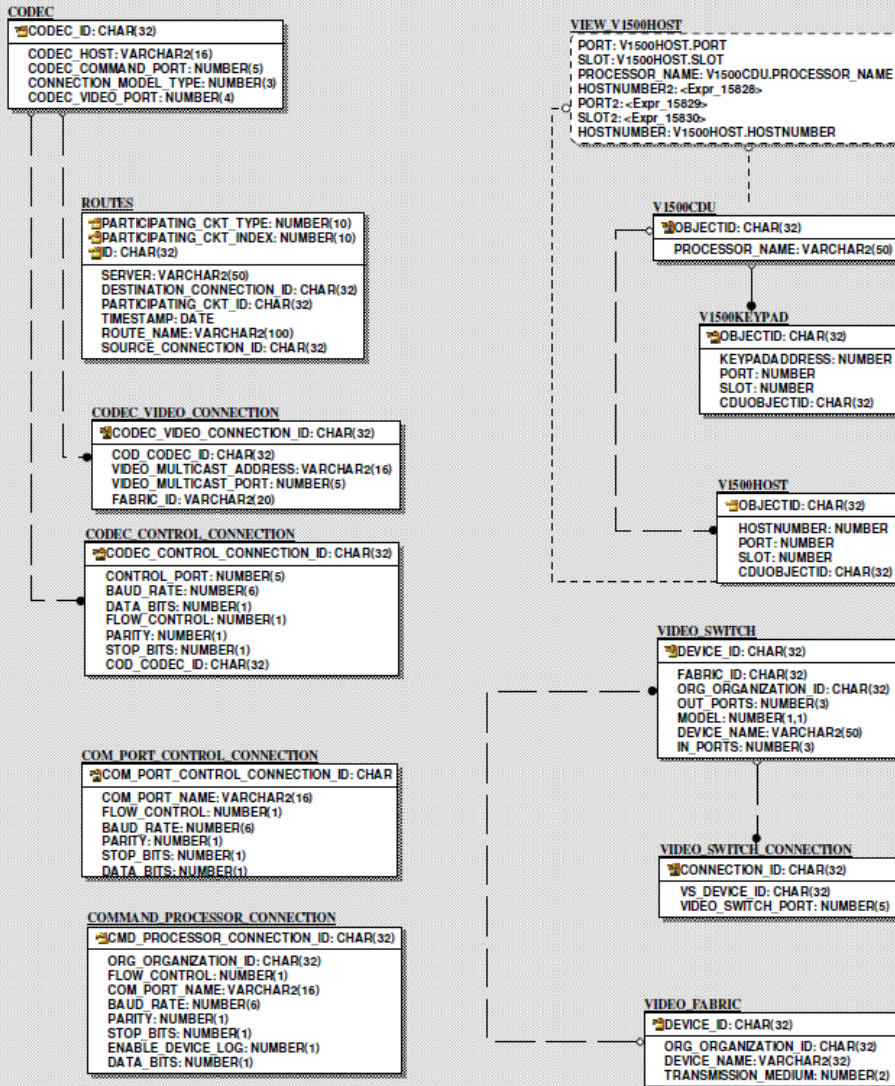




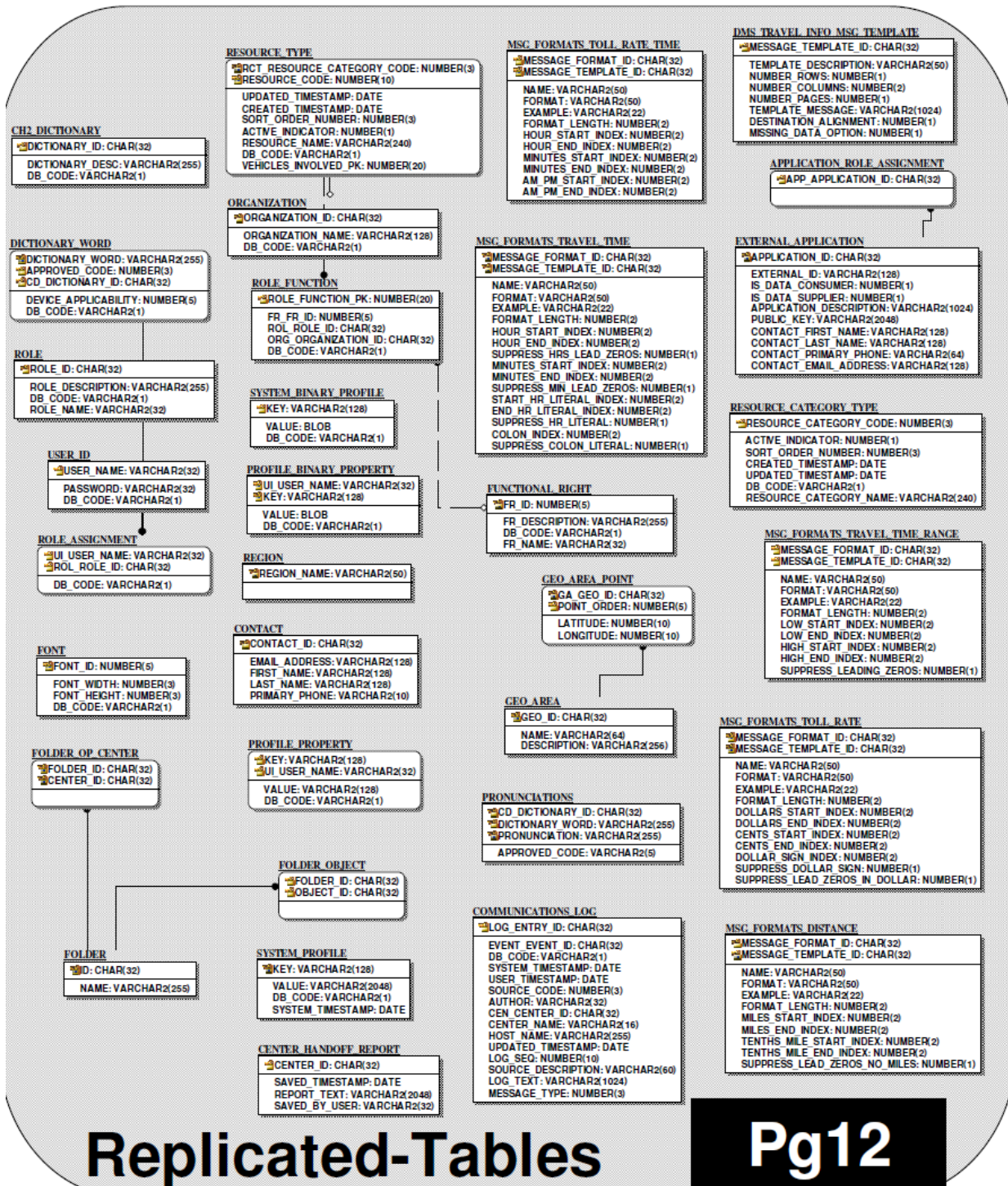
Pg09





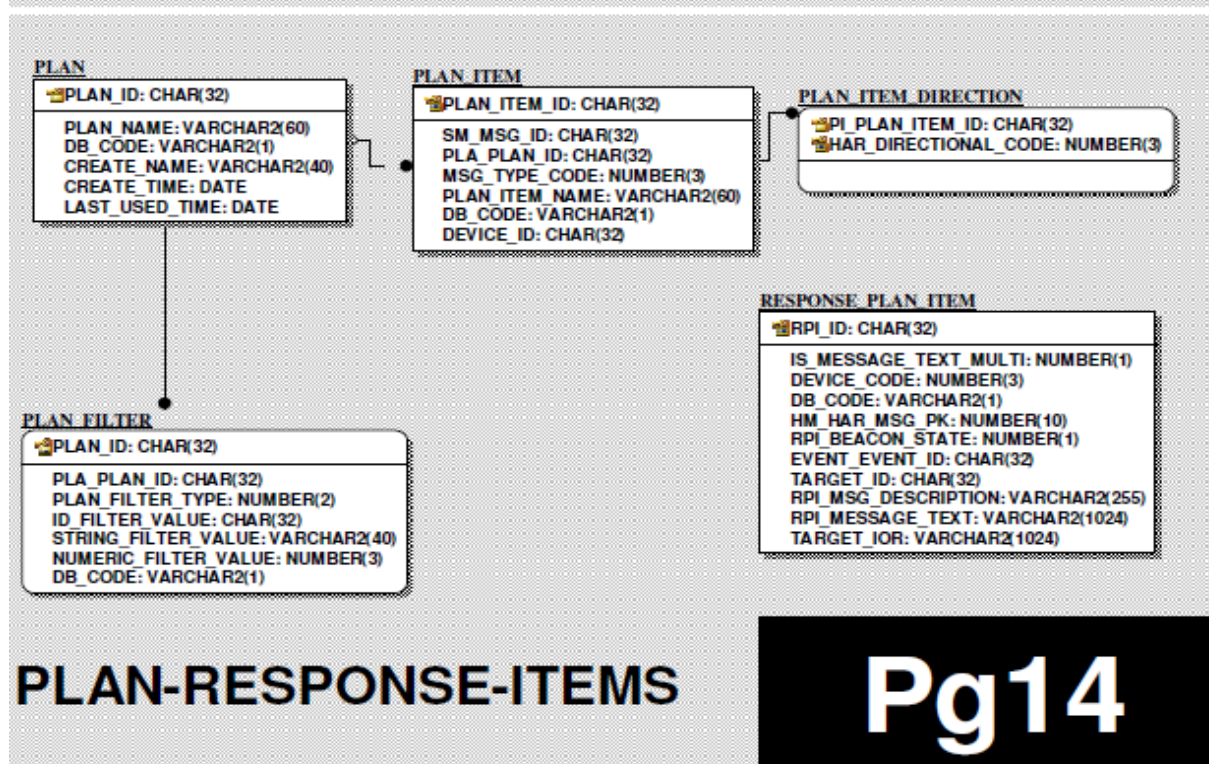
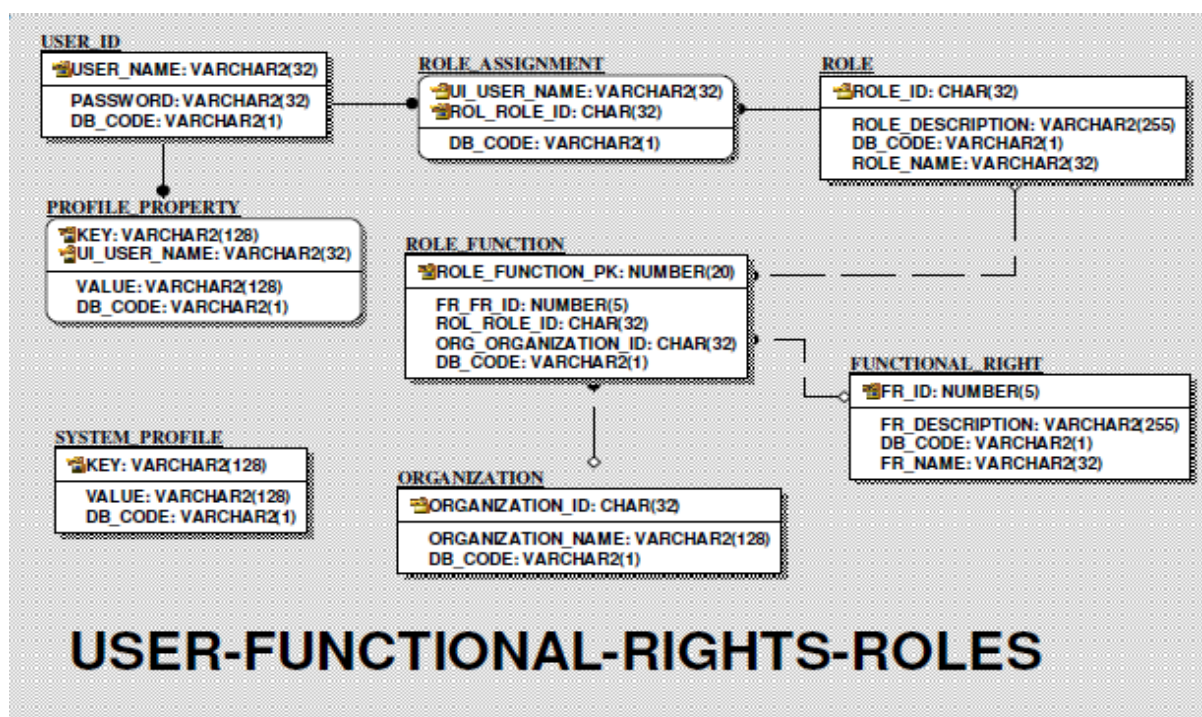












**Pg14**

Figure 2-4 R9 ERD

#### **2.4.1.1.2.2 Function to Entity Matrix Report**

The Create, Retrieve, Update, Delete (CRUD) matrix cross-references business functions to entities and shows the use of the entities by those functions. This report will be generated as part of the CHART O&M Guide.

#### **2.4.1.1.2.3 Table Definition Report –**

In existing tables shown below:

- Deleted columns/constraints marked with a minus sign (“-”)
- Modified columns/constraints marked with an asterisk (“\*”)
- New columns/constraints marked with a plus sign (“+”)

### **2.4.1.1.2.3.1 Tables Modified for the Decision Support Feature**

#### **2.4.1.1.2.3.1.1 CHART DB**

The R9 Decision Support feature will require 5 new tables being added to the CHART Oracle database. These tables will support Decision Support message template persistence, Decision Support disabled devices and plans, and local device proximity for an event.

##### **DS\_DMS\_MSG\_TEMPLATE Table:**

**Rights: This table will need full C/R/U/D rights for the MESSAGEUTILITYSERVICE user.**

This table stores the main data describing each Decision Support DMS message template in the system. It stores the unique ID for each template (primary key), a descriptive string, and the message text (MULTI with Decision Support tags embedded).

DS\_DMS\_MSG\_TEMPLATE Columns:

MESSAGE\_TEMPLATE\_ID CHAR(32) NOT NULL

TEMPLATE\_MESSAGE VARCHAR2(1024) NOT NULL

SHOW\_BEACONS NUMBER(1) NOT NULL

##### **DS\_MSG\_TEMPL\_FILTER\_TYPES Table:**

**Rights: This table will need full C/R/U/D rights for the MESSAGEUTILITYSERVICE user.**

This table is a lookup table that contains a row for each supported Decision Support message template filter type (as of R9: 1 - Supported Event Types, 2 - Supporter Distance Types, 3 - Supported Proximities, 4 - Max Columns supported).

DS\_MSG\_TEMPL\_FILTER\_TYPES Columns:

FILTER\_TYPE\_ID NUMBER(2) NOT NULL

DESCRIPTION VARCHAR2(50) NOT NULL

##### **DS\_MSG\_TEMPL\_FILTER Table:**

**Rights: This table will need full C/R/U/D rights for the MESSAGEUTILITYSERVICE user.**

This table stores the filter criteria for each template stored in the DS\_DMS\_MSG\_TEMPLATE table. For each message template there will be one or more rows in this table for each supported filter type in the DS\_MSG\_TEMPL\_FILTER\_TYPES table.

DS\_MSG\_TEMPL\_FILTER Columns:

MSG\_TEMPLATE\_ID CHAR(32) NOT NULL

FILTER\_TYPE\_IDNUMBER(2)NOT NULL  
EVENT\_TYPENUMBER(2)  
DIST\_TYPENUMBER(2)  
PROX\_IS\_SAME\_ROUTENUMBER(1)  
PROX\_IS\_UPSTREAMNUMBER(1)  
PROX\_DIR\_TYPENUMBER(2)  
MAX\_COLUMNSNUMBER(2)



**DS\_DISABLED\_SUGGESTION\_OBJ Table:**

**Rights: This table will need full C/R/U/D rights for the TRAFFICEVENTSERVICE user.**

This table stores a list of identifiers of CHART Objects that should no longer be suggested for use in response to a traffic event. It stores the unique ID for each device or plan and the unique ID for its related traffic event. These 2 IDs together will form the primary key. When a traffic event is deleted from the EVENT table, all related records in this table will be deleted.

DS\_DISABLED\_SUGGESTION\_OBJ Columns:

EVENT\_ID CHAR(32)NOT NULL

OBJECT\_ID CHAR(32)NOT NULL

**EVENT\_DEVICE\_PROX\_INFO Table:**

**Rights: This table will need full C/R/U/D rights for the TRAFFICEVENTSERVICE user.**

This table stores a list of devices that are located near a traffic event. It stores the unique ID for each device, the unique ID for its related traffic event, the device type, and the proximity information (same route, direction type, and upstream) for the device. The 2 IDs together will form the primary key. When a traffic event is deleted from the EVENT table, all related records in this table will be deleted.

EVENT\_DEVICE\_PROX\_INFO Columns:

DEVICE\_ID CHAR(32)NOT NULL

EVENT\_ID CHAR(32)NOT NULL

DEVICE\_TYPENUMBER(3)NOT NULL

SAME\_ROUTENUMBER(1)NOT NULL

DIR\_TYPENUMBER(3)NOT NULL

UPSTREAMNUMBER(1)NOT NULL

The device type will use the existing DEVICE values from the CODE\_LIST and CODE\_LIST\_ITEM tables. The direction type will require new values to be added to the CODE\_LIST and CODE\_LIST\_ITEM tables.

**CODE\_LIST Table:**

New Values:

<b>CODE_TYPE_NAME</b>
PROX_DIR_TYPE

**CODE\_LIST\_ITEM Table:**

New Values:

<b>CDL_CODE_TYPE_NAME</b>	<b>TYPE_CODE</b>	<b>TYPE_NAME</b>	<b>ACTIVE_INDICATOR</b>
PROX_DIR_TYPE	1	DIRECTION_SAME	1
PROX_DIR_TYPE	2	DIRECTION_OPPOSITE	1
PROX_DIR_TYPE	0	DIRECTION_UNKNOWN	1



## 2.4.1.1.2.3.2 Tables Modified for the Desktop Video feature

### 2.4.1.1.2.3.2.1 CHART DB

The CENTER and FLASH\_VIDEO\_STREAM\_CONTROL tables will be modified for the Desktop Video feature to add new columns. A new table: VIDEO\_SESSION will be added. These changes are described in more detail below:

#### CENTER Table

A MAX\_VIDEO\_SESSIONS column will be added to the CENTER table. This is marked with a plus sign (“+”) below:

Name	Null?	Type
-----	-----	-----
CENTER_ID	NOT NULL	CHAR(32)
CENTER_NAME	NOT NULL	VARCHAR2(16)
CREATED_TIMESTAMP		DATE
UPDATED_TIMESTAMP		DATE
DB_CODE		VARCHAR2(1)
MG_DEFAULT_MONITOR_GROUP_ID		CHAR(32)
CENTER_OWNING_ORGANIZATION		CHAR(32)
<b>+MAX_VIDEO_SESSIONS</b>		<b>NUMBER(4)</b>

## FLASH\_VIDEO\_STREAM\_CONTROL Table

An IS\_PUBLIC column will be added to the FLASH\_VIDEO\_STREAM\_CONTROL table. Also, a STREAM\_BLOCKED column will be added to record the last known blocking status of the stream. This column is nullable because the blocking status will not initially be known until a block or unblock command is issued. These columns are marked with a plus sign (“+”) below:

Name	Null?	Type
-----	-----	-----
CAMERA_DEVICE_ID	NOT NULL	CHAR (32)
FLASH_VIDEO_HOST	NOT NULL	VARCHAR2 (16)
PASSWORD		VARCHAR2 (32)
PORT	NOT NULL	NUMBER (5)
<b>+IS_PUBLIC</b>	<b>NOT NULL</b>	<b>NUMBER (1)</b>
+STREAM_BLOCKED		NUMBER (1)

The IS\_PUBLIC flag will default to a value of 1 for existing records, as all SFS configurations have been treated as “public” in CHART prior to R9.

The primary key for this table is (CAMERA\_DEVICE\_ID, FLASH\_VIDEO\_HOST, PORT) and will remain unchanged.

## VIDEO\_SESSION Table (New)

A new VIDEO\_SESSION table will be added to store video session information in case the managing CHART service is restarted. This table will not be replicated.

Name	Null?	Type
-----	-----	-----
SESSION_ID	NOT NULL	CHAR (32)
USER_NAME	NOT NULL	VARCHAR2 (32)
CENTER_ID	NOT NULL	CHAR (32)
CENTER_NAME	NOT NULL	VARCHAR2 (16)
CREATED_TIMESTAMP	NOT NULL	DATE
UPDATED_TIMESTAMP	NOT NULL	DATE
USER_HOST	NOT NULL	VARCHAR2 (50)
USER_IP	NOT NULL	VARCHAR2 (15)
CLIENT_APP_HOST	NOT NULL	VARCHAR2 (50)
CLIENT_INSTANCE_ID	NOT NULL	CHAR (32)
CAMERA_ID		CHAR (32)
TOUR_ID		CHAR (32)
SUBJECT_DESC	NOT NULL	VARCHAR2 (50)

CENTER\_ID will be a foreign key referencing CENTER.CENTER\_ID with cascade delete so that if an operations center is deleted, the video sessions for that center will also be deleted.

The primary key for this table is SESSION\_ID.

The CAMERA\_ID and TOUR\_ID columns are both nullable, but one of these will be specified. The CAMERA\_ID (if not null) references CAMERA.DEVICE\_ID and TOUR\_ID (if not null) references TOUR.TOUR\_ID.

### 2.4.1.1.2.3.3 Tables Modified for the G4 RTMS feature

#### 2.4.1.1.2.3.3.1 CHART DB

##### TSS Table

In the TSS table there will be one new column added, and the TSS\_MODEL\_ID column will be able to take a new value, a value of 3 meaning G4 RTMS. (Existing values of 1 meaning (X3) RTMS and 2 meaning External TSS will be maintained.) A new row will be added to the CODE\_LIST\_ITEM table to define the new value of 3 for CODE\_TYPE 'Model ID (TSS)'.

DEVICE_ID	NOT NULL VARCHAR2 (32)
TSS_MODEL_ID	NOT NULL NUMBER (5)
ORG_ORGANIZATION_ID	NOT NULL CHAR (32)
DB_CODE	VARCHAR2 (1)
DEVICE_NAME	NOT NULL VARCHAR2 (15)
DEVICE_LOCATION	VARCHAR2 (60)
DROP_ADDRESS	NOT NULL NUMBER (5)
INITIAL_RESPONSE_TIMEOUT	NOT NULL NUMBER (10)
DEFAULT_PHONE_NUMBER	VARCHAR2 (25)
POLL_INTERVAL_SECS	NOT NULL NUMBER (5)
PORT_TYPE	NUMBER (1)
PORT_MANAGER_TIMEOUT	NUMBER (10)
BAUD_RATE	NUMBER (6)
DATA_BITS	NUMBER (1)
FLOW_CONTROL	NUMBER (1)
PARITY	NUMBER (1)
STOP_BITS	NUMBER (1)
ENABLE_DEVICE_LOG	NOT NULL NUMBER (1)
CREATED_TIMESTAMP	DATE
UPDATED_TIMESTAMP	DATE
CEN_ALERT_CENTER_ID	CHAR (32)
EXT_ID_SYSTEM_ID	VARCHAR2 (35)
EXT_ID_AGENCY_ID	VARCHAR2 (35)
EXT_ID_TSS_ID	VARCHAR2 (256)
TCP_HOST	VARCHAR2 (16)
TCP_PORT	NUMBER (5)
MAINT_ORGANIZATION_ID	CHAR (32)
DISPLAY_BEARING	NOT NULL NUMBER (3)
+ENABLE_SCHEDULED_BIT_IND	NOT NULL NUMBER (1)

## TSS\_RAW\_DATA Table (NEW)

A new TSS\_RAW\_DATA table will be created to support logging of TSS raw traffic data directly to the live database. This table will be modeled to mirror the TSS\_RAW\_DATA table in the archive database, to provide for easy archiving of data from the live database to the archive database.

```
TSS_DEVICE_ID          NOT NULL VARCHAR2 (32)
SYSTEM_TIMESTAMP        NOT NULL DATE
ZONE_NUMBER            NOT NULL NUMBER (2)
ZONE_GROUP_NUMBER      NOT NULL NUMBER (2)
DIRECTION              NOT NULL NUMBER (3)
SPEED                 NOT NULL NUMBER (4, 1)
VOLUME                NOT NULL NUMBER (5)
OCCUPANCY             NOT NULL NUMBER (4, 1)

CONSTRAINT "TSS_RAW_DATA_PK"
  PRIMARY KEY ("TSS_DEVICE_ID", "SYSTEM_TIMESTAMP", "ZONE_NUMBER")
CONSTRAINT "TSS_RAW_DATA_FK"
  FOREIGN KEY (TSS_DEVICE_ID)
    REFERENCES TSS (DEVICE_ID) ON DELETE CASCADE
CONSTRAINT TRD_ZONE_NUMBER CHECK (ZONE_NUMBER BETWEEN 1 AND 12)
CONSTRAINT TRD_ZONE_GROUP_NUMBER CHECK (ZONE_GROUP_NUMBER BETWEEN 1 AND 12)
CONSTRAINT TRD_DIRECTION CHECK (DIRECTION IN (0, 2, 4, 6, 8, 254, 255))
CONSTRAINT TRD_OCCUPANCY CHECK (OCCUPANCY BETWEEN -1 AND 100)
```

## TSS\_ZONE\_GROUP and TSS\_ZONE Tables

The TSS\_ZONE\_GROUP and TSS\_ZONE tables will potentially have more rows: there may be up to 12 zone groups per TSS and up to 12 zones per zone group (maximum of 12 zones per TSS), as the G4 RTMS can support up to 12 zones. (The X3 RTMS can support up to 8 zones). There are no structural changes to support this.

### 2.4.1.1.2.4 PL/SQL Module Definition and Database Trigger Reports

There are no new PL/SQL modules for CHART R9.

### 2.4.1.1.2.5 Database Size Estimate - provides size estimate of current design

#### G4 RTMS Feature

The new TSS\_RAW\_DATA table of recent live TSS data is expected to hold a minimum of 30 minutes' worth up to a maximum of 4.5 hours' worth of raw data. Polls occur every five minutes so that is 6 polls to 30 polls per working online internal TSS in the live database at all times. The average number of polls stored at any given time then, is 18 rows per online internal TSS. A poll consists of data for every zone. There are 137 internal TSSs and 503 zones for those TSSs in CHART at this time. A TSS\_RAW\_DATA row is 100 bytes in CHART at this time. If all are online and working that is 503 x 18 x 100 bytes = 905,400 bytes, or just under 1MB additional

space needed in the database support TSS data, up to a peak every 4 hours (30 polls worth) of  $595 \times 30 \times 100 = 1,509,000$  bytes, or about 1.5MB. There is no change in the archive database for TSS – the same data will eventually get archived into the archive database the same as before.

#### **2.4.1.1.2.6 Data Distribution**

There are no changes to data distribution for R9.

#### **2.4.1.1.2.7 Database Replication**

There are no changes to database replication for R9.

#### **2.4.1.1.2.8 Archival Migration**

There are no changes to archival migration for R9.

#### **2.4.1.1.2.9 Database Failover Strategy**

There are no changes to the database failover strategy for R9.

#### **2.4.1.1.2.10 Reports**

No reports will be added or updated for R9. Since R5, the CHART reporting function has been transferred to University of Maryland.

### **2.4.1.2 CHART Flat Files**

The following describes the use of flat files in CHART.

#### **2.4.1.2.1 Service Registration Files**

There are no new Java services and therefore no new service registration files for CHART R9.

#### **2.4.1.2.2 Service Property Files**

Except as noted, there are no new service property files for CHART R9.

#### **2.4.1.2.3 GUI Property Files**

There are only minor updates to the GUI properties file in its WEB-INF directory for CHART R9.

#### **2.4.1.2.4 Arbitration Queue Storage Files**

There are no changes to Arbitration Queue Storage Files for R9.

#### **2.4.1.2.5 Device Logs**

There are no changes to Device Log Files for R9, except that TSS debug log files will now include G4 as well as X3 RTMS logs.

#### **2.4.1.2.6 Traffic Sensor Raw Data Logs**

Traffic Sensor Raw Data Log Files no longer be used starting in R9. TSS raw data will be logged directly to the live database (whence it will be archived to the archive database, on the order of every four hours). *Note: this section of the design document will be removed starting with R10; it will no longer be relevant.*

#### **2.4.1.2.7 Service Process Logs**

All CHART services write to a process log, used to provide a historical record of activity undertaken by the services. These logs are occasionally referenced by software engineering personnel to diagnose a problem or reconstruct a sequence of events leading to a particular anomalous situation. These logs are automatically deleted by the system after a set period of time defined by the service's properties file, so they do not accumulate infinitely. These files are stored in the individual service directories and are named by the service name and date, plus a ".txt" extension. These logs are typically read only by software engineering personnel. Except where noted, there are no changes for service process logs for R9 features.

#### **2.4.1.2.8 Service Error Logs**

All CHART services write to an error log, used to provide detail on certain errors encountered by the services. Most messages, including most errors, are captured by the CHART software and written to the process logs, but certain messages (typically produced by the Java Virtual Machine itself, by COTS, or DLLs) cannot be captured by CHART Software and instead are captured in these "catch-all" logs. Errors stored in these logs are typically problems resulting from a bad installation; once the system is up and running, errors rarely appear in these error logs. Debugging information from the JacORB COTS, which is not usually indicative of errors, can routinely be found in these error logs, as well. These log files can be reviewed by software engineering personnel to diagnose an installation problem or other type of problem. These logs are automatically deleted by the system after a set period of time defined by the service's properties file, so they do not accumulate infinitely. These files are stored in the individual service directories and are named by the service name and date, plus an ".err" extension. These logs are typically read only by software engineering personnel. Except where noted, there are no changes for service error logs for R9 features.

#### **2.4.1.2.9 GUI Process Logs**

Like the CHART background services, the CHART GUI service also writes to a process log file, used to provide a historical record of activity undertaken by the process. These GUI process logs are occasionally referenced by software engineering personnel to diagnose a problem or reconstruct a sequence of events leading to a particular anomalous situation. These logs are automatically deleted by the system after a set period of time defined by the GUI service's properties file, so they do not accumulate infinitely. These files are stored in the `chartlite/LogFiles/` directory under the `WebApps/` directory in the Apache Tomcat installation area. They are named by the service name ("chartlite") and date, plus a ".txt" extension. These logs are typically read only by software engineering personnel. Additional log files written by the Apache Tomcat system itself are stored in the `log/` directory in the Apache Tomcat installation area.

- R9 GUI changes do not change the way the GUI process logs operate.

#### **2.4.1.2.10 FMS Port Configuration Files**

The CHART Communications Services read a Port Configuration file, typically named `PortConfig.xml`, upon startup, which indicates which ports are to be used by the service and how they are to be initialized. A Port Configuration Utility is provided which allows for addition, removal of ports and editing of initialization parameters. As indicated by the extension, these files are in XML format. This means these files are hand-editable, although the Port Configuration Utility allows for safer, more controlled editing. The Port Configuration files are typically modified only by software engineers or telecommunications engineers.

- There are no changes to this section for the any of the R9 features.

#### **2.4.1.2.11 Watchdog Configuration Files**

There are no changes to the Watchdog configuration files for any of the R9 features.



## **2.4.2 Database Design**

Changes made to the CHART database design for Release 9 features are described below.

### **2.4.2.1 Decision Support**

#### **2.4.2.1.1 CHART DB**

The R9 Decision Support feature will require five new tables being added to the CHART Oracle database, as described above.

### **2.4.2.2 Desktop Video**

#### **2.4.2.2.1 CHART DB**

The changes to the database design for R9 are detailed above. Following is a description of these changes:

- Configuration of maximum video sessions allowed per operations center (column added to the CENTER table)
- Persistence of video sessions to handle the restarting of the CHART service managing the video sessions (new VIDEO\_SESSION table).
- Addition of a “public” flag to govern whether a camera’s stream will be blocked within an SFS if the Block (or Unblock) To Public command is issued, and a “stream blocked” flag to record the last known blocking status (columns added to FLASH\_VIDEO\_STREAM\_CONTROL table).
- Additional Database trigger will be design for DS\_MSG\_TEMPL\_FILTER table, this will check null values for the table.

### **2.4.2.3 G4 RTMS**

#### **2.4.2.3.1 CHART DB**

The changes to the database design for R9 are detailed above. Following is a description of these changes:

##### **TSS Table**

In the TSS table there will be one new column, but the TSS\_MODEL\_ID column will be able to take a new value, a value of 3 meaning G4 RTMS. (Existing values of 1 meaning (X3) RTMS and 2 meaning External TSS will be maintained.) A new row will be added to the CODE\_LIST\_ITEM table to define the new value of 3 for CODE\_TYPE ‘Model ID (TSS)’.

##### **TSS\_RAW\_DATA Table**

A new TSS Raw Data table will be created to support logging of TSS raw traffic data directly to the live database. This table will be modeled to mirror the TSS\_RAW\_DATA table in the archive database, to provide for easy archiving of data from the live database to the archive database.

#### **TSS\_ZONE\_GROUP and TSS\_ZONE Tables**

The TSS\_ZONE\_GROUP and TSS\_ZONE tables will potentially have more rows: there may be up to 12 zone groups per TSS and up to 12 zones per zone group (maximum of 12 zones per TSS), as the G4 RTMS can support up to 12 zones. (The X3 RTMS can support up to 8 zones). There are no structural changes to support this.)

#### **2.4.2.4 Archiving - Changes**

The CHART Archive database stores data from the CHART operational system as part of a permanent archive. The CHART Archive database design is a copy of the CHART operational system for those tables containing system, alert, traveler information messages and their underlying data, and event log information. In addition, the CHART Archive database stores detector data. We will be changing the code for TSS RAW DATA load, In R9 we will not be using sql loader to load the tss raw data. We will create new package to load the TSS raw data directly from the live database tables.

## 3 Key Design Concepts

---

### 3.1 Decision Support

In this release the focus of decision support is on helping the operator to determine the best DMS devices to use in response to a traffic event and to suggest messages that the operator should consider putting on the selected signs. To that end, the system can be pre-configured with message templates that pertain to one or more traffic event types, signs within certain proximities, and signs with specified geometries. Upon request, the system finds the signs near a traffic event (using further away devices as more lanes close) and then searches through the pre-configured templates looking for those that pertain to each sign for the traffic event. The variables in the template are then replaced with current data from the traffic event to create the suggested message. Each sign can have multiple suggested messages, so the system scores each message it creates from a template based on how specific the message content is (number of parameters, etc) and presents the suggestions with the highest scoring suggestion at the top.

In addition to DMS message suggestions, CHART R9 also enhances the traffic event response panel by notifying the user when the current response plan does not contain a DMS that decision support rules indicate should be used, or when they are using a DMS that the rules indicate should not be used. The page is also enhanced by allowing the operator to request a preview map that shows all response devices and what messages they would have on them if the response plan was executed.

#### 3.1.1 Key Design Decisions (Decision Support)

- All R9 Decision Support work is Traffic Event specific and relies heavily on the availability of current Traffic Event data in order to work. To optimize performance and reduce complexity it was decided to implement the R9 Decision Support algorithms in the CHART Traffic Event Service.
- Although the R9 Decision Support suggestions revolve around traffic events, it is not difficult to imagine wanting to create decision support type suggestions for other portions of the CHART system. To ensure this type of flexibility, the `DecisionSupportEnabled` interface has been defined outside of the traffic event domain. This CORBA interface can be implemented by any component that needs to be able to make suggestions.
- The types of data suggested have also been designed for easy extensibility so that the system can suggest actions other than “put this message on this DMS” or “use this plan”.
- The `DecisionSupportUtility` package has been added to contain decision support algorithms and utilities that are not traffic event specific. This will allow us to re-use this code for other decision support related activities in the future.
- Assembling a list of suggested actions for an operator based on current system conditions can be a relatively long running operation. The design takes advantage of the existing

CommandStatus interface used for device communications and camera control requests to allow the UI to continue on without waiting for the suggestions. The server then updates the status text to inform the user of the progress on their request. The interface is designed to allow the server to stream back suggestions as they are found if desired.

- The CHART Mapping GIS Web Service will be extended to allow the CHART system to request information about the nearest exit to the traffic event, and also to determine if each device that is located on the same route as the traffic event is upstream or downstream from the traffic event.
- Each Traffic Event will cache the list of devices that are nearby and whether that device is on the same route as the traffic event. If it is on the same route it will also cache flags that indicate if the device is in the same direction as the traffic event and if the device is upstream or downstream from the traffic event. This cache will be updated periodically on a configurable basis to account for the rare occasions when devices are added, deleted or moved. It will also be updated any time the traffic event location is changed. This cached information will allow the system to quickly provide a list of devices that are recommended for use in the response plan for the traffic event (upstream devices in same direction) and a list of devices that are not recommended for use (downstream devices in same direction).
- The R9 Decision Support DMS templates are designed as an extension of the DMS message template framework that was created for travel time and toll rate messages.

## 3.2 Desktop Video

### General Configuration

To display desktop video, the CHART GUI will make use of a camera's Flash video stream that has already been configured in one of the several Streaming Flash Servers (SFSs) used in the CHART system using the SFS software. (Also note that a camera stream must be configured in the Transcoding Server to encode the image for use by a Streaming Flash Server). Examples of SFSs include: Internal, Public, SWGI, and Mobile. A given camera may have streams set up in some or all of the SFSs. A camera must also be configured in CHART to specify which of the SFSs have streams representing it before CHART will support desktop video for that camera. In R9, each instance of the GUI will use **one** specific SFS, so a camera will only be viewable if it is configured for that SFS.

### Video Sessions: Limiting and Tracking Desktop Video Usage

The concept of a "video session" is being introduced in R9 to track and limit usage of desktop video. A video session represents the potential use of the network resources to stream a single live camera image, and can be thought of as a single video window that is open on a user's desktop. In an attempt to limit the network bandwidth usage by desktop video, the number of video sessions will be limited per operations center, so that each operations center will support a maximum total number of video sessions that all users logged into that center can have open. Before opening each video window, the system will check to make sure that the resource limit

has not been exceeded. Once the initial resource limit check is made, the user can stream video without checking the resource limit again as long as that video window is open (regardless of whether video is actually streaming or not). An administrator will be able to end a user's video session to free up system resources, which will cause the video to stop streaming. Video sessions also provide status tracking information, and users will be able to see who is viewing which cameras and tours via desktop video. If for some reason a user's browser or the GUI servlet fails to release the video session when a video window is closed, there will be resource cleanup logic at both the GUI servlet level and at the CHART service level to make sure resources are released after a configurable timeout. If operations center's desktop video limit is reduced below the current usage, all current sessions will continue however no new sessions can be opened until the total for that operations center is below the new limit.

Video sessions will be managed by the Resources Module (User Management Service) rather than by the Video Service, to simplify logic and reduce potential failure conditions that could occur. It might seem like video sessions should be managed by the Video Service, but what they really represent is network usage and they are completely independent of all of the types of objects hosted by the Video Service. Video sessions are really a negotiation between the GUI and Resource Management, as it is the GUI that is using the network resources and the Video Service does not need to be involved.

## **Blocking / Unblocking and Video Stream Status**

The Block / Unblock Camera To Public functionality that has blocked a camera's streams in all SFSs configured in CHART since R5 will be modified in R9 to accommodate the non-public SFSs that will be added to support desktop video streaming. (Prior to R9, only public SFSs were configured in CHART, and the only reason SFSs were configured in CHART was to support blocking.) Non-public SFSs will not be affected by these "public" commands, although in R9 a camera's stream will be blockable / unblockable via a separate command on an individual SFS basis (regardless of whether the SFS is public or not).

Currently the SFS software API does not support querying the existence or the blocking status of the camera streams within the SFS. As a result, in R9 CHART will maintain its own status information for each camera stream within an SFS. To maintain the accuracy of this status, it is assumed (see Assumptions/Constraints) that blocking of a camera's stream will ONLY be done via CHART, and the SFS software will not be used for this purpose. Should the status get out of synch, the workaround (using CHART) would involve issuing a block (or unblock) command to the individual SFS to get it back to a known state.

Blocking a camera to a given organization will affect desktop video usage for the organization associated to the operations center at which the user logged in. If the matching organization is blocked, the user will not be able to view desktop video.

NOTE – Since R5 the SFS configuration settings have been configured in the System Profile, but these System Profile settings only act as a template from which values are copied into the camera configuration when the SFS is added to a camera. This makes the camera configurations independent of the System Profile settings, but changing settings such as the SFS IP address, password, or name (or in R9, the public flag) would have to be done for each camera in the system if the setting is changed via the CHART GUI (although it could be changed more easily via a database update followed by a restart). Changing the design to have each camera reference the settings in the System Profile would make it easier to change the SFS configuration settings. However, because it would be a non-trivial change and the issue existed prior to R9, this document does not attempt to change the nature of the previous design, although an enhancement PR (LevC3159) has been entered to consider changing it if time allows.

## **Video Tours**

A user will be able to view preconfigured (persistent) video tours via desktop video, with a couple of limitations. First, only cameras that have video streams configured for the GUI's assigned SFS will be included in the tour. Second, any camera presets configured for the tour will be ignored. (The use of presets would require camera communications, and multiple users viewing tours with presets would have the potential to cause serious communication problems with the cameras. Also, using presets with desktop video would have required precise timing between the application of the preset and the desktop video buffering in the GUI, which may not even be feasible.)

The video from multiple cameras in the tour will be buffered one at a time, so that there will be a slight delay when switching between camera streams. (This was shown in the prototype and was deemed to be acceptable.) Because buffering is done one camera at a time, viewing a tour will count as a single video session with respect to the operations center resource limit.

Normally only one instance of a camera is allowed on a desktop however desktop tours are permitted to contain the same camera as an existing desktop session – even if it is the only camera displaying in the desktop tour.

## **Camera Control**

Prior to R9, the system required a camera to be displayed on a local monitor before the user was allowed to request control of the camera. In R9, this restriction will be removed or relaxed. If the camera is not already displayed on a local monitor when the user requests control and if the camera is eligible for streaming, a desktop video session will automatically be opened for the user so that they can see the effects of the control operations on the camera.

Similarly, if a camera control session contains the camera's video and that same camera is later put on a local monitor, the user is then allowed to remove the desktop video session from the camera control session.

As it is with local monitors, for the purposes of camera control, a camera in a desktop tour does not satisfy the requirement that a user be able to view a camera before obtaining control. This is true even if the camera is the only one displaying in the desktop tour.

### 3.3 G4 RTMS

This task is to provide support for the G4 RTMS Traffic Sensor and to support multi-drop deployment of TSS devices. Currently SHA owns G4 RTMS models of TSS but operates them in X3 compatibility mode because the CHART software does not support the G4 protocol. The G4 protocol provides information for 12 detection zones instead of 8 zones like the X3. The G4 protocol is substantially different than the X3; it is built to support additional traffic parameters (such as gap (a.k.a. headway)), up to 6 vehicle classes, additional zones, and future expansion. The CHART software will be changed to allow the administrator to specify whether a TSS is a X3 RTMS (the “original” RTMS model) or a new G4 RTMS model. When a G4 model is indicated, zone groups defined for the TSS will be able to use zones 1 through 12. When a G4 model is polled, volume, speed, and occupancy data will be retrieved for each of the zones in use. The CHART GUI will be updated to allow display of 12 zones for G4 RTMS sensors in each place where 8 zones can currently be displayed. All existing rules regarding display of summary data vs. actual data will apply to the additional zones.

The data exporter will be changed as necessary to support detectors with 12 zones. The current TSS simulation software built into the TSS service will be updated to support simulating data with up to 12 zones from a G4 RTMS.

The logging of the raw traffic data will be updated to log directly to the live CHART database instead of using flat files. This will be done for both the X3 and G4 models and will support 8 or 12 zones of data. Up until R8, traffic data is logged to flat files, and is loaded from there into the archive database once a day by a database job. The current job used to pull raw data from flat files into the archive database will be retired and the archiving task will be updated to archive the RTMS raw data directly from the live CHART database(s). An advantage of logging to the database is it will make it easier to depersist the latest poll data into each TSS during startup of the TSS Service. In order to facilitate this, actually two tables will be used: one called TSS\_RAW\_DATA which is strictly for the database archive job to archive and purge. The TSS Service will never read this table, so after archiving the archive job is free to purge everything in this table without prejudice. The TSS\_RAW\_DATA\_LATEST table will contain only the most recent datapoint for each TSS (that is, when a new datapoint for a TSS comes in, the previous datapoint will be deleted first, then the new datapoint inserted. This eliminates the risk that a broken archive job could leave thousands of records in the table the TSS Service reads, which would slow down startup of the TSS Service, and greatly simplifies the depersistence code.

The TSS Service will no longer run Built-in Test (BIT) based on the receipt of a bad health bit during data polling. The G4 RTMS does not populate this health bit, and the manufacturer has recommended against this practice even for the X3 RTMS. For one thing, for both models, running BIT halts data collection for about 20 seconds (vehicles are not detected). As a compromise, the system can be configured to poll TSSs automatically once a night. A system-wide TSS BIT execution time (expected to be during early morning hours, e.g., 3:00 in the morning) can be configured into the System Profile. Each TSS can be individually configured to run or not run BIT at the scheduled time. It is expected that only flaky detectors will be



configured to run BIT, and most normally operating detectors will not, but that is a client decision.

To support multi-drop capability, the CHART software will be changed to group the polling of TSSs that use TCP/IP communications and have the same IP address and TCP port, or the same phone number. During the polling cycle, each detector that shares the same connection or phone number will be polled sequentially to avoid having these detectors polled simultaneously, which is known to cause contention issues. During this polling process, detectors will be checked to see if they are configured to run scheduled BIT, and if the test has not been run since the BIT time arrived. Any detectors needing to run BIT will run their BIT following data collection.

## 3.4 Error Processing

In general, CHART traps conditions at both the GUI and at the server. User errors that are trapped by the GUI are reported immediately back to the user. The GUI will also report communications problems with the server back to the user. The server may also trap user errors and those messages will be written to a server log file and returned back to the GUI for display to the user. Additionally, server errors due to network errors or internal server problems will be written to log files and returned back to the GUI.

## 3.5 Packaging

### 3.5.1 CHART

This software design is broken into packages of related classes. The table below shows each package that is new or changed to support the Release 9 features.

Package Name	Package Description
<b>CHART2.CameraControlModule</b>	This package is changed for R9 to support blocking and unblocking SFS streams for a camera, and keeping track of the stream blocking status.
<b>CHART2.Common</b>	This package has been changed by adding some new data types that can be used to describe the proximity of a location to another location. This will be used by decision support to describe the location of devices in relation to a traffic event that is being responded to.
<b>CHART2.DecisionSupport</b>	This CORBA package has been added in R9. It contains new interfaces and data types that allow client components to request decision support generated suggestions from server components.
<b>CHART2.DecisionSupportUtility</b>	This Utility package contains server side utility code that can be used by future components that implement the DecisionSupportEnabled interface. The intent of this package is to increase reuse by keeping common code out of the TrafficEventModule package.
<b>CHART2.DMSUtility.templates</b>	This package has been updated to re-factor the existing travel times and toll rates templates to provide a common framework that can be used for decision

Package Name	Package Description
<b>CHART2.DMSUtility.templates.DecisionSupport</b>	support templates as well. This package contains the DMS template code that is decision support specific.
<b>CHART2.DMSUtility.templates.TravelerInfo</b>	This package contains the DMS template code that is travel time and toll rate specific.
<b>CHART2.MessageTemplateManagement</b>	This package has been changed by altering the IDL structures and interfaces that are used for message templates to be extensible in order to support the new template types needed for decision support.
<b>CHART2.MessageTemplateModule</b>	This package is being modified to use the new common templates framework and to allow it to support the current travel time, toll rate, and decision support DMS templates as well as to make it flexible enough to easily support decision support HAR templates in the future.
<b>CHART2.ResourcesModule</b>	This package is changed for R9 to manage user video sessions, which are limited per operations center to limit bandwidth usage.
<b>CHART2.TrafficEventModule</b>	This package is being modified in order to allow the TrafficEventGroup to suggest possible DMS messages that the operator should consider using based on current event details and nearby devices.
<b>CHART2.TSSControlModule</b>	This package is changed for R9 to support the the G4 RTMS.
<b>CHART2.TSSManagement</b>	This CORBA package will be modified for R9 to add support for G4 RTMS in addition to X3 RTMS (formerly known simply as RTMS).
<b>CHART2.Utility.objectcache.planmgmt</b>	This package contains the classes necessary in order to allow the object cache to discover and maintain a cache of ProxyPlan and ProxyPlanItem classes for use in decision support.
<b>CHART2.Utility.objectcache.templategmt</b>	This package contains the classes necessary in order to allow the object cache to discover and maintain a cache of message template classes for use in decision support.
<b>chartlite.data</b>	This package is changed for R9 to add a video session wrapper class and to modify the operations center wrapper object to keep track of video sessions.
<b>chartlite.data.decisionsupport</b>	This package contains GUI data classes for decision support.
<b>chartlite.data.templates</b>	This GUI data package is modified to make templates more generic in order to support the new template types needed for decision support.
<b>chartlite.data.trafficevent</b>	This GUI data package is modified to support the new decision support functionality that is traffic event specific.
<b>chartlite.data.tss</b>	This CORBA package will be modified for R9 to add support for G4 RTMS in addition to X3 RTMS (formerly known simply as RTMS).
<b>chartlite.data.video</b>	This package is changed for R9 to support flash stream blocking status and to add a “public” flag.
<b>chartlite.servlet</b>	This package is changed for R9 to add some miscellaneous utility methods to the servlet classes to support desktop video.
<b>chartlite.servlet.dms</b>	This GUI package is modified to support the processing needed to allow users to manage DMS related decision

Package Name	Package Description
<b>chartlite.servlet.templates</b>	support message templates. This GUI package is modified to support the new generic base template classes used by decision support, travel time and toll rate templates.
<b>chartlite.servlet.trafficevents</b>	This GUI package is modified to support the new processing that is needed to allow the user to request DMS message suggestions for traffic event response as well as to allow the user to view a traffic event response preview map.
<b>chartlite.servlet.tss</b>	This package is changed for R9 to support the the G4 RTMS.
<b>chartlite.servlet.usermgmt</b>	This GUI package has been modified to support the new system profile configuration settings needed for decision support.
<b>chartlite.servlet.video</b>	This package is changed for R9 to support desktop video-related requests.
<b>Flex: video</b>	This package is new for R9 to support video streaming via an Adobe Flex control.

## **3.6 Assumptions and Constraints**

### **3.6.1 Decision Support**

1. The Decision Support feature will suggest messages only for CHART DMS devices in R9. It will not suggest messages for external DMS devices (those that are not under direct control of the CHART DMS Service). This should be a safe assumption because CHART operators currently do not have any means of activating messages on external DMS devices.
2. The CHART system does not currently have the ability to determine if there is an easily traversable route between a DMS and a traffic event. Further, during the JADs it was determined that we do not want to pay for the use of a subscription API. Therefore, R9 Decision Support will suggest messages for all DMS devices within the configured radius of the traffic event. Note: an administrator will be able to indicate if the system should suggest messages only for devices on the same route as the traffic event.

### **3.6.2 Desktop Video**

1. Camera streams will not be blocked within the SFS software except through the CHART GUI. Blocking of the camera streams using the SFS software or any other method will result in CHART's status for a camera's stream being incorrect. (As of R9 the SFS API does not support querying the stream status, so there is no way for CHART to query the existence or blocked/unblocked status of a camera's streams within the SFS server).

### **3.6.3 G4 RTMS**

1. For RTMS we will continue to use only the traffic parameters currently used in CHART for the X3 model (volume, speed, occupancy). We will not add other data made available in the G4 model such as gap, headway, and vehicle classifications.
2. For RTMS we will continue to use only status values currently used in CHART (hardware failed, comm failed, or OK) and will not add other status data to CHART that is available from the G4, such as Side Fired vs. Forward Looking and voltage.
3. For RTMS technical support will continue to be available from the G4 manufacturer (ISS).
4. For RTMS the G4 device will be provided by the Radio Shop will continue to be available and accessible from CSC and over the Internet.
5. For RTMS support for multi-drop will be done without IDL changes. We will move the polling timers out of the PolledTSSImpl object to a higher level (the factory) so that we can share threads amongst detectors that have the same IP and port, therefore serializing access to them.

6. For RTMS the multi-dropped TSSs will be hosted within the same TSS service/factory.
7. For RTMS we will not update the stand alone RTMS simulator.
8. For RTMS the data exporter will handle detectors with 12 zones without changes. The data exporter will need to be tested however.

## 4 Use Cases – Decision Support

The use case diagrams depict new functionality for CHART R9 Decision Support and also identify existing features that will be enhanced. The use case diagrams for Decision Support exist in the Tau design tool in the Release 9 area. The sections below indicate the title of the use case diagrams that apply to Decision Support.

### 4.1.1.1 RespondToTrafficEvent (Use Case Diagram)

The system allows an operator to control devices in response to an event through the use of a response plan. The user may add devices to the plan, select the desired state of the devices, then activate the plan. Any of the devices used by the event response plan may be deactivated while the event is open by removing the item for that device from the plan. When the event is closed, if the response plan is active, it will be deactivated automatically.

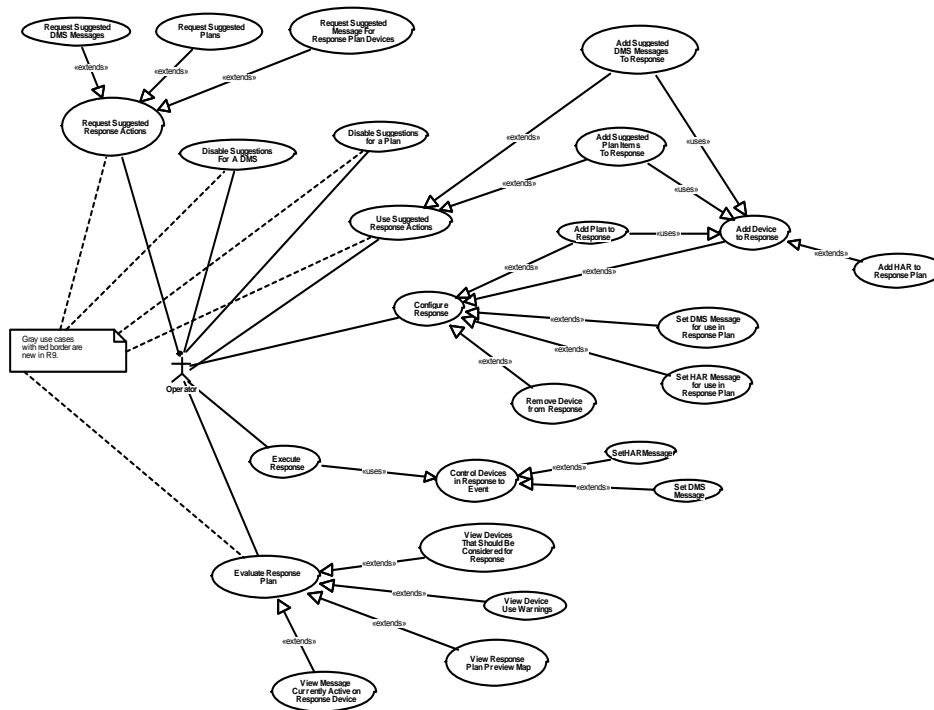


Figure 4-1. RespondToTrafficEvent (Use Case Diagram)

#### 4.1.1.1.1 Add Device to Response (Use Case)

An operator with the correct functional rights may add a device to the response plan of a traffic event. Please refer to the extending use cases to see the types of devices that can be added to a response plan.



#### **4.1.1.1.2 Add HAR to Response Plan (Use Case)**

An operator with the correct functional rights may add a HAR to the response plan of a traffic event. Doing so will cause the message on that HAR to be set when the response plan is executed. In this version of the system, the HAR will be added with a blank initial message which the operator may modify before executing the plan. In future releases, the HAR object could suggest a message based on the properties of the traffic event whose response plan it was being added to.

#### **4.1.1.1.3 Add Plan to Response (Use Case)**

An operator with the correct functional rights may add the items from an existing plan to a traffic event's response plan. This action is performed by dragging the plan to the traffic event. Doing so will cause the traffic event's response plan to contain a response item for each item in the plan. This feature will be useful when responding to recurring congestion events or special events where a predefined plan can be established and stored. The operator may also drag one or more individual plan items from a plan to a traffic event. If response items already exist in the response plan for the devices referenced by the plan items, the message will be changed in the response item; otherwise, new response items will be created. These changes or additions of the response items are only proposed changes and will not affect the device unless the response items are subsequently executed.

#### **4.1.1.1.4 Add Suggested DMS Messages To Response (Use Case)**

A user may opt to add a suggested DMS message to the current response plan. If the device is already in the response plan, it will be updated to use the suggested message. The user may also opt to activate the message immediately when adding it.

#### **4.1.1.1.5 Add Suggested Plan Items To Response (Use Case)**

A user may opt to add one or more suggested DMS plan item messages to the current response plan. If the device is already in the response plan, it will be updated to use the suggested message. The user may also opt to activate the message immediately when adding it.

#### **4.1.1.1.6 Configure Response (Use Case)**

An operator with the correct functional rights may configure a response plan to control devices in response to a traffic event. Please refer to the extending use cases for details.

#### **4.1.1.1.7 Control Devices in Response to Event (Use Case)**

A user may control devices in response to an event by adding the devices to the traffic event response plan, editing a message for each device, then activating the response plan items.

#### **4.1.1.1.8 Disable Suggestions For A DMS (Use Case)**

A user with sufficient privileges will be able to disable suggestions for a particular DMS. The suggestions can be disabled for the life of the traffic event, or may be disabled only for the current response plan suggestions session. Once they are disabled, DMS suggestions may also be re-enabled.

#### **4.1.1.1.9 Disable Suggestions for a Plan (Use Case)**

A user with sufficient privileges will be able to disable suggestions for a particular Plan. The suggestions can be disabled for the life of the traffic event, or may be disabled only for the current response plan suggestions session. Once they are disabled, DMS suggestions may also be re-enabled.

#### **4.1.1.1.10 Evaluate Response Plan (Use Case)**

Users may utilize the system to evaluation their current response plan. Refer to the extending use cases for a list of supported evaluation tools.

#### **4.1.1.1.11 Execute Response (Use Case)**

An operator with the correct functional rights may execute the response plan for a particular traffic event. Performing this operation will place the message from each response plan item on the arbitration queue of the corresponding device.

#### **4.1.1.1.12 Remove Device from Response (Use Case)**

An operator with the correct functional rights may remove a device from a traffic event's response plan. This action will result in the message from this response plan item being removed from the arbitration queue of the device which is being removed.

#### **4.1.1.1.13 Request Suggested DMS Messages (Use Case)**

The system will suggest template based DMS messages for DMS devices that are located within a configurable distance of the traffic event. Each device can have multiple template based suggestions. Each template is tested to verify that it is suitable for the traffic event type, distance and proximity of the sign to the traffic event, and sign geometry. Each suitable template will result in a suggested message. Suggestions are scored according to their specificity of the DMS/Traffic Event combination. Thus, a template that pertains only to incidents will be scored higher than a similar template that pertains to all traffic event types. Templates with higher scores are display to the user first. The system will also locate plans that contain the nearby DMS devices and will also suggest the planned messages for each of these devices. Because the system has no knowledge of the content of these messages they will be given the lowest possible score.

#### **4.1.1.1.14 Request Suggested Message For Response Plan Devices (Use Case)**

A user with sufficient privileges may request message suggestions for either all DMSs in

the response plan or DMSs in the response plan with a blank message. The system will suggest messages for a DMS in the response plan even if that DMS was not originally suggested for this traffic event.

#### **4.1.1.1.15 Request Suggested Plans (Use Case)**

The system shall also suggest plans that contain DMS devices that are within the configurable range of the traffic event. Each plan that contains any "nearby" devices will be suggested. Plans will be scored according to the percentage of total plan items that are nearby DMS devices and will be presented to the user in order of score.

#### **4.1.1.1.16 Request Suggested Response Actions (Use Case)**

A user with sufficient privileges may request the suggested response actions for a traffic event. Refer to extending use cases to see what types of suggestions are supported.

#### **4.1.1.1.17 Set DMS Message (Use Case)**

The message on a DMS can be set when the DMS is online or in maintenance mode. When the DMS is online, the message is set by the DMS's arbitration queue. This queue sets the message of the DMS to be the message that is on the queue that has the highest priority. When the DMS is in maintenance mode, an operator with proper functional rights can set the message on a DMS directly.

#### **4.1.1.1.18 Set DMS Message for use in Response Plan (Use Case)**

An operator with the correct functional rights may modify the message which will be displayed on a DMS when a traffic event response plan is executed. This can be done by using the message editor, or dragging a stored DMS message to the item in the event response.

#### **4.1.1.1.19 Set HAR Message for use in Response Plan (Use Case)**

An operator with the correct functional rights may modify the message which will be broadcast from a HAR when the traffic event's response plan is executed. This can be done using the HAR message editor, or dragging a HAR stored message to the item in the response plan.

#### **4.1.1.1.20 SetHARMessage (Use Case)**

A HAR's message is set through the execution of an event response plan or set directly by an administrator when the device is in maintenance mode. The message activation may specify messages which were previously stored in message slots in the controller or a message that was created using the HAR message editor.

When activating a HAR message created by the message editor the user may choose to use

the default header and trailer or just use the message body for the entire message. Messages activated in this manner shall be loaded into the HAR controller in the slot designated for immediate broadcast.

A HAR message activation also specifies if each associated SHAZAM should be activated or not. The selected notifiers will be activated only after the message has been activated on the HAR.

The system shall support sending messages to at least 4 HARs at one time; each constituent of a synchronized HAR counts as 1 HAR toward this total. A synchronized HAR is comprised of individual constituent HARs that play the same message at the same time. Each constituent can be specified as being active or inactive; messages activated on a synchronized HAR are only activated on the active constituents.

#### **4.1.1.1.21 Use Suggested Response Actions (Use Case)**

A user with sufficient privileges may use a suggested response action. Refer to the extending use cases to see the list of supported response actions the user may use.

#### **4.1.1.1.22 View Device Use Warnings (Use Case)**

The system will highlight any devices that are currently in the response plan that appear to be inappropriate. A good example of a potentially inappropriate device is a DMS that is on the same route, same direction as the traffic event but is downstream from the event.

#### **4.1.1.1.23 View Devices That Should Be Considered for Response (Use Case)**

A user will be able to see a list of devices that decision support rules indicate should be considered for use in the response plan of the traffic event but that are currently in the response plan.

#### **4.1.1.1.24 View Message Currently Active on Response Device (Use Case)**

The system will allow a user to see the currently active message on a response device. This can aid the user in deciding whether the currently planned response message is useful.

#### **4.1.1.1.25 View Response Plan Preview Map (Use Case)**

A user may view a map that shows the traffic event, all currently planned response plan items, and any response plan items that should be considered. For each response plan item, the suggested message can be viewed by opening the device callout. Devices in the response plan that should not be considered will be highlighted on the map. Devices not in the response plan that should be considered will also be highlighted on the map. This will help the operator to visualize the future state of the system if the response plan is executed as currently planned.

#### 4.1.1.2 ConfigureDecisionSupport (Use Case Diagram)

The system allows an operator to configure the decision support settings that are used to generate suggested actions (both devices and messages) for a traffic event response plan. The user may configure general settings (distances, proximities, and route types), DMS message templates, route direction replacement text, event type and incident type tag replacements, and word substitutions.

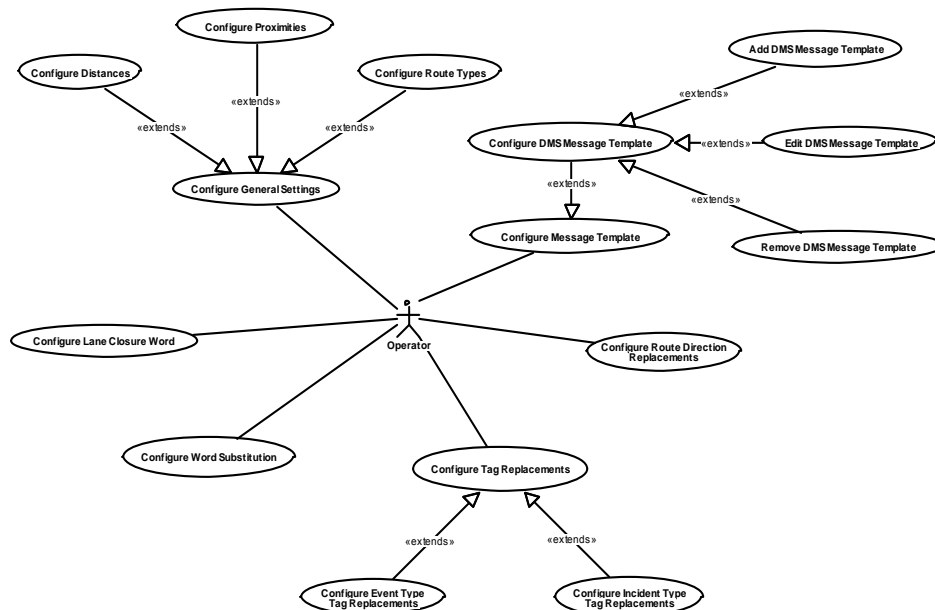


Figure 4-2. ConfigureDecisionSupport (Use Case Diagram)

##### 4.1.1.2.1 Add DMS Message Template (Use Case)

A user with sufficient privileges may create a new DMS message template. The user must configure a message consisting of static text and parameter tags, applicable traffic event types, applicable proximities, applicable distances, and applicable geometries. The possible parameter tags include: event type, incident type, route type, route number, route name, route direction, intersecting exit number, intersecting exit road name, and lane closures. The resulting message can contain up to 2 pages. The applicable geometries must support at least 8 characters per line and at least 2 lines per page.

##### 4.1.1.2.2 Configure Distances (Use Case)

A user with sufficient privileges may configure the decision support distance settings. A user may configure the 3 maximum distances (in miles) - one for each of the 3 distance types: Immediate, Near, and Far. A user may configure a lane closure percentage for each of the 3 distance types. These settings are used to determine whether a DMS should be suggested for a traffic event.

#### **4.1.1.2.3 Configure DMS Message Template (Use Case)**

A user with sufficient privileges may configure DMS message templates. These templates are used to generate message suggestions for a DMS. Refer to the extending use cases for the details of what can be configured for a message template.

#### **4.1.1.2.4 Configure Event Type Tag Replacements (Use Case)**

A user with sufficient privileges may configure the decision support event type tag replacement settings. The event type replacements are used to replace event type values from the traffic event that are replacing event type tags in the message template. A user may configure a long version and/or a short version for each event type.

#### **4.1.1.2.5 Configure General Settings (Use Case)**

A user with sufficient privileges may configure the general settings for decision support including: distance, proximity, and route type. These settings are used when determining if a DMS or DMS Message Template should be suggested for a traffic event. Refer to the extending use cases for the details of what can be configured.

#### **4.1.1.2.6 Configure Incident Type Tag Replacements (Use Case)**

A user with sufficient privileges may configure the decision support incident type tag replacement settings. The incident type replacements are used to replace incident type values from the traffic event that are replacing incident type tags in the message template. A user may configure a long version and/or a short version for each incident type.

#### **4.1.1.2.7 Configure Lane Closure Word (Use Case)**

A user with sufficient privileges may configure the decision support lane closure word substitution settings. Both a long and a short substitution can be configured for each word. The lane closure words are used to replace lane closure tags in the message template. The lane closure word defaults to a value of "CLOSED" for Planned Roadway Closures and Special Events and a default value of "BLOCKED" for all other traffic event types.

#### **4.1.1.2.8 Configure Message Template (Use Case)**

A user with sufficient privileges may configure decision support message templates. These templates are used to generate decision support message suggestions. Refer to the extending use cases for the details of what can be configured for a message template.

#### **4.1.1.2.9 Configure Proximities (Use Case)**

A user with sufficient privileges may configure the decision support proximity settings. A user may configure the applicable proximities using the following 3 options: a) "Same route only?", b) "Same direction only?", and c) "Upstream only?". The same direction and upstream options will only apply if the same route option is true. These settings are used to determine whether a DMS should be suggested for a traffic event.



#### **4.1.1.2.10 Configure Route Direction Replacements (Use Case)**

A user with sufficient privileges may configure the decision support route direction replacements settings. The replacements are used to replace directions from the traffic event that are replacing tags in the message template. A user may specify a blank replacement for a route direction to indicate that the route direction should not be used in message content.

#### **4.1.1.2.11 Configure Route Types (Use Case)**

A user with sufficient privileges may configure the decision support route type settings. A user may configure the Route Type / Number and Route Name separately for each available route type. The route type settings will determine if a template will be suggested for a DMS based on whether a Route Type / Number tag or a Route Name tag exists in a DMS message template.

#### **4.1.1.2.12 Configure Tag Replacements (Use Case)**

A user with sufficient privileges may configure the decision support tag replacement settings. These settings are used when replacing values from the traffic event that are replacing tags in the message template. Refer to the extending use cases for the details of what can be configured.

#### **4.1.1.2.13 Configure Word Substitution (Use Case)**

A user with sufficient privileges may configure the decision support word substitutions settings. Both a long and a short substitution can be configured for each word. It is valid for the user to enter blanks (not text) for short version, long version, or both. If both the long and short versions are blank, the word or phrase will be removed from the message completely. The substitutions are not used to replace text entered by the user.

#### **4.1.1.2.14 Edit DMS Message Template (Use Case)**

A user with sufficient privileges may edit an existing DMS message template. The user may edit the current values for the message, applicable traffic event types, applicable proximities, applicable distances, and applicable geometries.

#### **4.1.1.2.15 Remove DMS Message Template (Use Case)**

A user with sufficient privileges may remove an existing DMS message template.

## 5 Detailed Design – Decision Support

### 5.1 Human-Machine Interface

#### 5.1.1 Decision Support Features

This section describes the features of Decision Support that are available to a user for R9. Many of these features can be seen in the Response Plan section of the Traffic Event Details page (see **Error! Reference source not found.** below) including: the ability to request suggested DMSs for the Traffic Event, the ability to request suggested messages for a DMS already in the response plan, an indication of a DMS in the response plan that is not recommended, an indication of a DMS that is not in the response plan that is recommended, and the ability to preview the response plan on a map.

**Add Items To Response Plan**

Search: All DMSs, HARs and Plans For: Search

Suggest DMSs Select DMSs Select HARs Select Plan

**Response Plan**

Target Device	Proposed Message	Status
SIM DMS 11 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	<a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>
SIM DMS 22 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	INCIDENT EISENHOWER MEM HWY ALL LANES OPEN <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	<a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>
SIM DMS 23 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	<a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>
SIM DMS 8 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	INCIDENT I-270 N BE ALERT <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	<a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>

**Suggest Message** All Blank Only **Edit DMS (Auto)** All Multiple **Edit DMS (Manual)** All Multiple **Execute** All Multiple **Revoke Execution** All Multiple **Remove** All Multiple

The DMS highlighted in red is not recommended.

[View the 2 Additional Recommended DMSs You Should Consider Using](#)

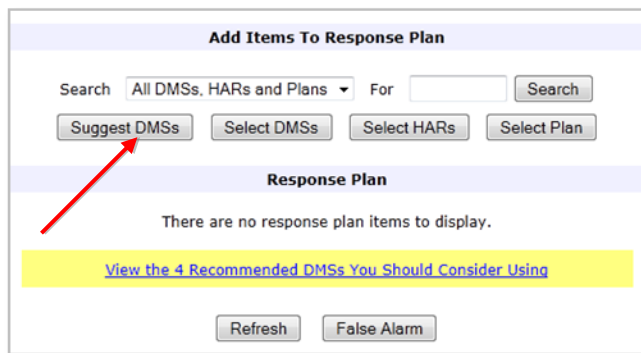
[Preview on Map](#) [Refresh](#) [False Alarm](#)

[General Info](#) [Incident Info](#) [Roadway Conditions](#) [Participation](#) [Response](#) [Notification](#) [Event History](#) [Summary](#) [Associated Events](#)

Figure 5-1. Traffic Event Response Plan Showing Features of Decision Support.

### 5.1.1.1 Viewing Suggested DMSs

A user can request suggested DMSs from the Response Plan section of the Traffic Event details page by clicking on the Suggest DMSs button (see Figure 5-2 below). Note: The button text will be changed to Suggest during implementation. DMSs will be suggested based on the following criteria: the type of traffic event, the distance of a DMS from the traffic event, the proximity of a DMS to the traffic event, the geometry of the DMS, the number of pages required for the template message, and the tags in the message template. Many of these factors are configurable and will be further described in section 5.1.2 below. The resulting suggested DMSs will be categorized based on their proximity to the traffic event into 2 groups: Upstream Devices and Other Devices.



**Figure 5-2. Requesting Suggested DMSs for a Traffic Event Response Plan.**

### 5.1.1.2 Upstream Devices

Upstream devices (see Figure 5-3 below) include only those DMSs that have a proximity to the traffic event of Upstream. These DMSs may be encountered by a driver on the same route as the traffic event in the same direction prior to encountering the traffic event. For each suggested DMS, the following information is included: the name and location, the proximity and distance, the suggested message, and the source and reason. The distance is the drivable distance from the DMS to the traffic event (if the drivable distance cannot be determined, actual distance between the devices will be displayed). Each suggested message will have a source (either template generated or the plan name) and the reasons that template message was suggested.

**Suggested Response Actions**

**Suggested Items**

DMSs (10)
Plans (3)

**Upstream Devices**

Target Device	Proximity	Suggested Message(s)	
<input type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 8</a> I-270 NORTH AT MP 7.5	Upstream Distance: 0.47 miles	<input checked="" type="radio"/>	<b>Message</b> 
		<b>Source / Reason</b> Template Generated: 4 Parameter(s), 5 Filter(s)	
<input type="checkbox"/> <b>Point-to-Point DMS</b> <a href="#">SIM DMS 22</a> I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE	Upstream Distance: 1.01 miles	<input checked="" type="radio"/>	<b>Message</b> 
		<b>Source / Reason</b> Template Generated: 3 Parameter(s), 3 Filter(s)	
		<a href="#">Show Additional Messages (3)</a>	
<input type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 21</a> I-270 NORTH AT EXIT 5 MD 189 FALLS RD	Upstream Distance: 2.43 miles	<input checked="" type="radio"/>	<b>Message</b> 
		<b>Source / Reason</b> Template Generated: 3 Parameter(s), 3 Filter(s)	
		<a href="#">Show Additional Messages (6)</a>	
<input type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 20</a> I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD	Upstream Distance: 3.76 miles	<input checked="" type="radio"/>	<b>Message</b> 
		<b>Source / Reason</b> Template Generated: 3 Parameter(s), 3 Filter(s)	
		<a href="#">Show Additional Messages (6)</a>	

**Other Devices** [\(Show\)](#)

☐ Permanently

**Figure 5-3. Suggested Upstream Devices and Messages for a Traffic Event Response Plan.**

A DMS may have no suggested messages if no applicable templates or plans exist, or it may have multiple suggested messages if multiple applicable templates or plans exist. If a suggested DMS contains more than one suggested message, only the most applicable message will be shown. A user may click on the Show Additional Messages link to view the additional suggested messages (see Figure 5-4 below).

DMSs (10)

Plans (3)

Upstream Devices

Target Device	Proximity	Suggested Message(s)	
<div> <div></div> <div>DMS</div> <div>SIM DMS 8</div> <div>I-270 NORTH AT MP 7.5</div> </div>	Upstream Distance: 0.47 miles	<div> <div></div> <div>Message</div> <div>INCIDENT I-270 N BE ALERT</div> </div>	Source / Reason Template Generated: 4 Parameter(s), 5 Filter(s)
<div> <div></div> <div>Plan 1 DMS</div> <div>SIM DMS 22</div> <div>I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE</div> </div>	Upstream Distance: 1.01 miles	<div> <div></div> <div>Message</div> <div>INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN</div> </div> <div> <div></div> <div>Message</div> <div>INCIDENT EISENHOWER MEM HWY ALL LANES OPEN</div> </div> <div> <div></div> <div>Message</div> <div>INCIDENT EISENHOWER MEM HWY N USE CAUTION</div> </div> <div> <div></div> <div>Message</div> <div>HIGH WATER PLEASE USE CAUTION</div> </div>	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)  Template Generated: 3 Parameter(s), 2 Filter(s)  Template Generated: 3 Parameter(s), 2 Filter(s)  Plan: AOC NORTH-RIPKEN STADIUM
<div> <div></div> <div>DMS</div> <div>SIM DMS 21</div> <div>I-270 NORTH AT EXIT 5 MD 189 FALLS RD</div> </div>	Upstream Distance: 2.43 miles	<div> <div></div> <div>Message</div> <div>INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN</div> </div>	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
Show Additional Messages (6)			
<div> <div></div> <div>DMS</div> <div>SIM DMS 20</div> <div>I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD</div> </div>	Upstream Distance: 3.76 miles	<div> <div></div> <div>Message</div> <div>INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN</div> </div>	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
Show Additional Messages (6)			

Additional Suggested Messages

**Figure 5-4. Additional Suggested Messages for a Suggested DMS.**



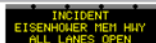


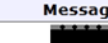


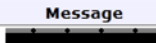






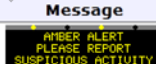
### 5.1.1.3 Other Devices

A user can click on the Show Other Devices link (see Figure 5-5 below) to view the other suggested devices.

Other Devices [\(Show\)](#)

**Figure 5-5. The Link to View the Details of Other Devices.**

Other devices (see Figure 5-6 below) include those DMSs that have a proximity to the traffic event of Same Route, Same Route/Same Direction, Same Route/Opposite Direction, Downstream, Other Route, No Direct Route, or Undetermined. These DMSs may be encountered by a driver on the same route as the traffic event in the same direction, on the same route as the traffic event in the opposite direction, or on another route if the driver were to leave that route and enter the route that the traffic event is located on. For each suggested DMS, the same information as described in section 5.1.1.2 is included.

Other Devices (Hide)			
Target Device	Proximity	Suggested Message(s)	
 <a href="#">SIM DMS 24</a> I-270 SOUTH AT EXIT 9	Opposite Dir, Downstream Distance: 1.34 miles	 	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
		<a href="#">Show Additional Messages (2)</a>	
 <a href="#">Close DMS 1</a> Within 1 Mile	Same Route Distance: 7.35 miles	 	Source / Reason Plan: +Chris SOC plan
 <a href="#">SIM DMS 26</a> I-370 WEST AT INDUSTRIAL DR	Other Route Distance: 1.18 miles	 	Source / Reason Template Generated: 4 Parameter(s), 2 Filter(s)
 <a href="#">SIM DMS 11</a> MONTGOMERY COUNTY MD	Undetermined Distance: 1.85 miles	No Suggested Messages	
 <a href="#">Close DMS 10</a> Within 10 Miles	Undetermined Distance: 2.75 miles	 	Source / Reason Plan: +Chris SOC plan
 <a href="#">DMS Test 2</a> MD 124 AT GOSHEN RD	Undetermined Distance: 3.61 miles	 	Source / Reason Plan: AMBER ALERT-SHA

**Figure 5-6. Suggested Other Devices and Messages for a Traffic Event Response Plan.**

#### 5.1.1.4 Viewing Additional Information for a Suggested DMS

##### Current DMS Message

A preview of the current message for each suggested DMS is available by hovering the mouse pointer over the name of the DMS (see Figure 5-7 below). If the DMS does not have a current message, the message will be blank.

 <a href="#">SIM DMS 22</a> I-270 NORTH PAST MONTGOMERY AVE	Upstream Distance: 1.01 miles	 	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
		<a href="#">Show Additional Messages (3)</a>	
 <a href="#">SIM DMS 21</a> I-270 NORTH AT EXIT 5 MD 189 FALLS RD	Upstream Distance: 2.43 miles	 	Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
		<a href="#">Show Additional Messages (6)</a>	

**Figure 5-7. DMS Current Message Preview on the Suggested Response Actions Page.**

##### Source /Reason Details

Templates are suggested based on the tags they contain and a set of configurable factors. Each suggested message will include information about the source and reason the message was suggested. More detailed information is available by hovering the mouse pointer over the Source / Reason text. A popup will display (see Figure 5-8 below) that shows both the tags that were included in the template and the filters that were used to determine that the message should be suggested.



Message		Tag Name	Count	
INCIDENT I-270 N BE ALERT	Template	Event Type	1	(s), 5
	Filter(s)	Route Name	1	
		Lane Closures	1	
INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN	Template	Filter Name	Count	(s), 3
	Filter(s)	Event Type	2 of 8	
INCIDENT EISENHOWER MEM HWY ALL LANES OPEN	Template	Proximity	1 of 12	(s), 2
	Filter(s)	Distance	3 of 3	
INCIDENT EISENHOWER MEM HWY N USE CAUTION	Template	Geometry	3	(s), 2
	Filter(s)	Pages	1	
HIGH WATER PLEASE USE CAUTION	Plan: AOC NORTH-RIPKEN STADIUM			

Figure 5-8. Suggested Message Scoring Details.

### 5.1.1.5 Selecting DMSs and Suggested Messages

Any DMS that has at least one suggested message may be selected and added to the response plan for the traffic event. The DMS can be selected by checking the checkbox next to its name (see Figure 5-9 below). The selected message for the DMS can be selected by checking the radio button next to the message. Only one suggested message can be selected per suggested DMS.

DMSs (10)
Plans (3)

Upstream Devices

Target Device	Proximity	Suggested Message(s)
<input checked="" type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 8</a> I-270 NORTH AT MP 7.5	Upstream Distance: 0.47 miles	<input checked="" type="radio"/> <b>Message</b> INCIDENT I-270 N BE ALERT Source / Reason Template Generated: 4 Parameter(s), 5 Filter(s)
<input checked="" type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 22</a> I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE	Upstream Distance: 1.01 miles	<input type="radio"/> <b>Message</b> INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
		<input checked="" type="radio"/> <b>Message</b> INCIDENT EISENHOWER MEM HWY ALL LANES OPEN Source / Reason Template Generated: 3 Parameter(s), 2 Filter(s)
		<input type="radio"/> <b>Message</b> INCIDENT EISENHOWER MEM HWY N USE CAUTION Source / Reason Template Generated: 3 Parameter(s), 2 Filter(s)
		<input type="radio"/> <b>Message</b> HIGH WATER PLEASE USE CAUTION Source / Reason Plan: AOC NORTH-RIPKEN STADIUM
<input type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 21</a> I-270 NORTH AT EXIT 5 MD 189 FALLS RD	Upstream Distance: 2.43 miles	<input checked="" type="radio"/> <b>Message</b> INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)
<input type="checkbox"/> <b>DMS</b> <a href="#">SIM DMS 20</a> I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD	Upstream Distance: 3.76 miles	<input checked="" type="radio"/> <b>Message</b> INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN Source / Reason Template Generated: 3 Parameter(s), 3 Filter(s)

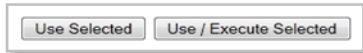
Other Devices [\(Show\)](#)

Use Selected
Use / Execute Selected
Remove Selected
☐ Permanently
Cancel

Figure 5-9. Selecting Devices to be Added to the Response Plan.

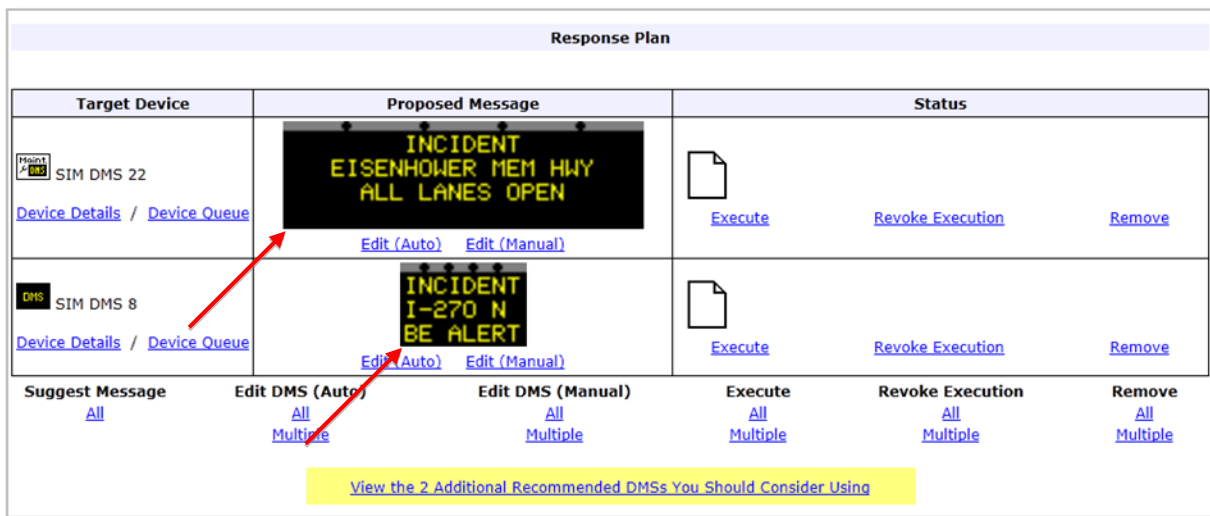
### 5.1.1.6 Adding DMSs to the Response Plan





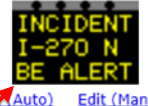

With at least one DMS (and its message) selected, a user can click on the Use Selected button (see Figure 5-10 below) to add the selected DMSs to the response plan.



**Figure 5-10. Buttons for Adding and Executing DMSs in the Response Plan.**

The selected DMSs will be added to the response plan with their suggested message (see Figure 5-11 below), but they will not be executed. A user can click on the Use / Execute Selected button to not only add the selected DMSs to the response plan but also execute them as well.



Response Plan					
Target Device	Proposed Message	Status			
 SIM DMS 22 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>			
 SIM DMS 8 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>			
<a href="#">Suggest Message</a> <a href="#">All</a>	<a href="#">Edit DMS (Auto)</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Edit DMS (Manual)</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Execute</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Revoke Execution</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Remove</a> <a href="#">All</a> <a href="#">Multiple</a>
<a href="#">View the 2 Additional Recommended DMSs You Should Consider Using</a>					

**Figure 5-11. Selected Devices Added to the Response Plan with Suggested Messages.**

### 5.1.1.7 Removing DMSs and Suggested Messages from the Suggestion List

A user may choose to remove a DMS and all of its suggested messages from the suggestion list. With at least one DMS selected, a user can click on the Remove Selected button (see Figure 5-12 below) to remove the selected DMSs from the suggestion list. If the Permanently checkbox is checked, the DMS (and its suggested messages) will be removed from the suggestion list until the traffic event is closed or until the user re-activates suggestions for that DMS. If the Permanently checkbox is not checked, the DMS (and its suggested messages) will be removed only from the current page. The DMS will be suggested again the next time any user requests suggestions for the traffic event.



**Figure 5-12. The Button for Removing DMSs and Suggested Messages from the Suggestions List.**

### 5.1.1.8 Viewing the Suggested Plan DMSs

When a user requests suggested DMSs from the Response Plan section of the Traffic Event details page (see Figure 5-2 above), each plan that contains one or more of the suggested DMSs will also be displayed. The Plan DMSs can be viewed by clicking on the Plan tab at the top of the Suggested Response Actions page (see Figure 5-13 below). For each Plan DMS, the proximity and distance to the traffic event is displayed. Each DMS also has a single message that comes from the plan.

DMSs (10)

Plans (3)

Plan

[AOC NORTH-RIPKEN STADIUM](#)  
Last used: N/A

Type

ANY

Items






Show / Hide Plan Items (5)

	Description	DMS	Proximity	Message
<input type="checkbox"/>	SIM DMS 20 - HIGH WATER PLEASE	<div>DMS</div> <a href="#">SIM DMS 20</a> I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD	Upstream Distance: 3.76 miles	<div>HIGH WATER PLEASE USE CAUTION</div>
<input type="checkbox"/>	SIM DMS 21 - HIGH WATER PLEASE	<div>DMS</div> <a href="#">SIM DMS 21</a> I-270 NORTH AT EXIT 5 MD 189 FALLS RD	Upstream Distance: 2.43 miles	<div>HIGH WATER PLEASE USE CAUTION</div>
<input type="checkbox"/>	SIM DMS 22 - HIGH WATER PLEASE	<div>Point / DMS</div> <a href="#">SIM DMS 22</a> I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE	Upstream Distance: 1.91 miles	<div>HIGH WATER PLEASE USE CAUTION</div>
<input type="checkbox"/>	DMS ABC123 - HIGH WATER PLEASE	<div>DMS</div> <a href="#">DMS ABC123</a> I-270 NORTH BETWEEN QUINCE ORCHARD RD AND MIDDLEBROOK RD	Downstream Distance: 3.47 miles	<div>HIGH WATER PLEASE USE CAUTION</div>
<input type="checkbox"/>	SIM DMS 23 - HIGH WATER PLEASE	<div>DMS</div> <a href="#">SIM DMS 23</a> I-270 NORTH AT EXIT 8 SHADY GROVE RD	Downstream Distance: 0.41 miles	<div>HIGH WATER PLEASE USE CAUTION</div>

Figure 5-13. Suggested Plans for a Traffic Event Response Plan.

### 5.1.1.9 Selecting Plan DMSs

Any Plan DMS may be selected and added to the response plan for the traffic event. The DMS can be selected by checking the checkbox next to its name (see Figure 5-14 below).

DMSs (10) Plans (3)					
Plan	Type	Items			
<a href="#">AOC NORTH-RIPKEN STADIUM</a> Last used: N/A	ANY	<a href="#">Show / Hide Plan Items (5)</a>			
		<input checked="" type="checkbox"/>	<b>Description</b>	<b>DMS</b>	<b>Proximity</b>
		<input type="checkbox"/>	SIM DMS 20 - HIGH WATER PLEASE	 <a href="#">SIM DMS 20</a> I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD	Upstream Distance: 3.76 miles
		<input type="checkbox"/>	SIM DMS 21 - HIGH WATER PLEASE	 <a href="#">SIM DMS 21</a> I-270 NORTH AT EXIT 5 MD 189 FALLS RD	Upstream Distance: 2.43 miles
		<input type="checkbox"/>	SIM DMS 22 - HIGH WATER PLEASE	 <a href="#">SIM DMS 22</a> I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE	Upstream Distance: 1.01 miles
		<input checked="" type="checkbox"/>	DMS SIM 123 - HIGH WATER PLEASE	 <a href="#">DMS SIM 123</a> I-270 NORTH BETWEEN QUINCE ORCHARD RD AND MIDDLEBROOK RD	Downstream Distance: 3.47 miles
		<input type="checkbox"/>	SIM DMS 23 - HIGH WATER PLEASE	 <a href="#">SIM DMS 23</a> I-270 NORTH AT EXIT 8 SHADY GROVE RD	Downstream Distance: 0.41 miles

**Figure 5-14. Selecting Plan Devices to be Added to the Response Plan.**

#### 5.1.1.10 Adding Plan DMSs to the Response Plan

If one or more DMS plan item is selected, the user can click on the Use Selected button (see Figure 5-10 above) to add the selected DMSs to the response plan. The selected DMSs will be added to the response plan with their plan message, but they will not be executed. A user can click on the Use / Execute Selected button to not only add the selected DMSs to the response plan but also execute them as well.

#### 5.1.1.11 Removing Plan DMSs from the Suggestion List

A user may choose to remove a Plan DMS from the suggestion list. With at least one DMS selected, a user can click on the Remove Selected button (see Figure 5-12 above) to remove the selected DMSs from the suggestion list. If the Permanently checkbox is checked, the suggested Plan DMS will be removed from the suggestion list until the traffic event is closed or until the user re-activates suggestions for that DMS. If the Permanently checkbox is not checked, the suggested Plan DMS will be removed only from the current view. This implies that the plan will be suggested again the next time any user requests suggestions for the traffic event.

#### 5.1.1.12 Getting Suggestions for Additional Recommended DMSs

If there are suggested devices with a proximity of Upstream that have not been added to the response plan, a warning will appear (see Figure 5-15 below) at the bottom of the response plan section of the traffic event details page.

[View the 2 Additional Recommended DMSs You Should Consider Using](#)

**Figure 5-15. The Link to View Additional Recommended DMSs.**

A user can click on this link to see the upstream devices that are not in the response plan (see Figure 5-16 below).

Upstream Devices

<input type="checkbox"/>	Target Device	Proximity	Suggested Message(s)	
<input type="checkbox"/>	<div><div>DMS</div><div><a href="#">SIM DMS 21</a></div><div>I-270 NORTH AT EXIT 5 MD 189 FALLS RD</div></div>	Upstream Distance: 2.43 miles	<div><div><div></div><div>INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN</div></div></div>	Template Generated: 3 Parameter(s), 3 Filter(s)
<a href="#">Show Additional Messages (6)</a>				
<input type="checkbox"/>	<div><div>DMS</div><div><a href="#">SIM DMS 20</a></div><div>I-270 NORTH AT EXIT 4 MD 927 MONTROSE RD</div></div>	Upstream Distance: 3.76 miles	<div><div><div></div><div>INCIDENT AHEAD EISENHOWER MEM HWY ALL LANES OPEN</div></div></div>	Template Generated: 3 Parameter(s), 3 Filter(s)
<a href="#">Show Additional Messages (6)</a>				





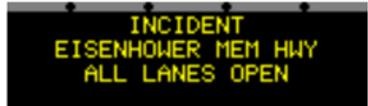


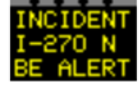

Other Devices

There are no DMSs in the Other Devices group.

**Figure 5-16. Additional Recommended DMSs Suggested for Response Plan.**

### 5.1.1.13 Getting Suggested Messages for DMSs in the Response Plan




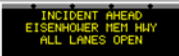
With at least one DMS in the response plan, a user can request message suggestions for either all DMSs in the response plan or DMSs in the response plan with a blank message.

Response Plan					
Target Device	Proposed Message	Status			
 SIM DMS 11 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>			
 SIM DMS 22 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>			
 SIM DMS 8 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>			
<b>Suggest Message</b> <a href="#">All</a> <a href="#">Blank Only</a>	<b>Edit DMS (Auto)</b> <a href="#">All</a> <a href="#">Multiple</a>	<b>Edit DMS (Manual)</b> <a href="#">All</a> <a href="#">Multiple</a>	<b>Execute</b> <a href="#">All</a> <a href="#">Multiple</a>	<b>Revoke Execution</b> <a href="#">All</a> <a href="#">Multiple</a>	<b>Remove</b> <a href="#">All</a> <a href="#">Multiple</a>
<a href="#">View the 2 Additional Recommended DMSs You Should Consider Using</a>					


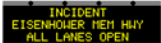
**Figure 5-17. Getting Suggested Messages for DMSs in the Response Plan.**

A user can click on the All link to get suggested messages for all DMSs in the response plan even if the DMS was not originally suggested for this traffic event (see Figure 5-18 below).

Upstream Devices

<input type="checkbox"/>	Target Device	Proximity	Suggested Message(s)	
<input type="checkbox"/>	 <a href="#">SIM DMS 8</a> I-270 NORTH AT MP 7.5	Upstream Distance: 0.47 miles	<input type="checkbox"/>	<div>Message</div> <div></div> <div>Source / Reason</div> <div>Template Generated: 4 Parameter(s), 5 Filter(s)</div>
<input type="checkbox"/>	 <a href="#">SIM DMS 22</a> I-270 NORTH PAST EXIT 6 MD 28 W MONTGOMERY AVE	Upstream Distance: 1.01 miles	<input type="checkbox"/>	<div>Message</div> <div></div> <div>Source / Reason</div> <div>Template Generated: 3 Parameter(s), 3 Filter(s)</div>
<a href="#">Show Additional Messages (3)</a>				

Other Devices

<input type="checkbox"/>	Target Device	Proximity	Suggested Message(s)	
<input type="checkbox"/>	 <a href="#">SIM DMS 11</a> MONTGOMERY COUNTY MD	Undetermined Distance: 1.85 miles	<input type="checkbox"/>	<div>Message</div> <div></div> <div>Source / Reason</div> <div>Template Generated: 3 Parameter(s), 2 Filter(s)</div>
<a href="#">Show Additional Messages (2)</a>				

**Figure 5-18. Suggested Messages for All DMSs in the Response Plan.**

#### 5.1.1.14 Warning About DMSs That Are Not Recommended

If the response plan contains at least one DMS with a proximity of downstream (Downstream, Opposite Direction/Upstream, or No Direct Route), a warning will appear (see Figure 5-19 below) at the bottom of the response plan section of the traffic event details page. In the list of response plan items, each DMS with a proximity of downstream will be highlighted (with color and with an icon ).

Response Plan				
Target Device	Proposed Message	Status		
<a href="#">SIM DMS 11</a> <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>		<a href="#">Execute</a>	<a href="#">Revoke Execution</a> <a href="#">Remove</a>
<a href="#">SIM DMS 22</a> <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>		<a href="#">Execute</a>	<a href="#">Revoke Execution</a> <a href="#">Remove</a>
<a href="#">SIM DMS 23</a> <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>		<a href="#">Execute</a>	<a href="#">Revoke Execution</a> <a href="#">Remove</a>
<a href="#">SIM DMS 8</a> <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>		<a href="#">Execute</a>	<a href="#">Revoke Execution</a> <a href="#">Remove</a>
<a href="#">Suggest Message</a> <a href="#">All</a> <a href="#">Blank Only</a>	<a href="#">Edit DMS (Auto)</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Edit DMS (Manual)</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Execute</a> <a href="#">All</a> <a href="#">Multiple</a>	<a href="#">Revoke Execution</a> <a href="#">All</a> <a href="#">Multiple</a> <a href="#">Remove</a> <a href="#">All</a> <a href="#">Multiple</a>
The DMS highlighted in red is not recommended.				
<a href="#">View the 2 Additional Recommended DMSs You Should Consider Using</a>				

**Figure 5-19. Response Plan Showing Warning About a DMS That is Not Recommended.**



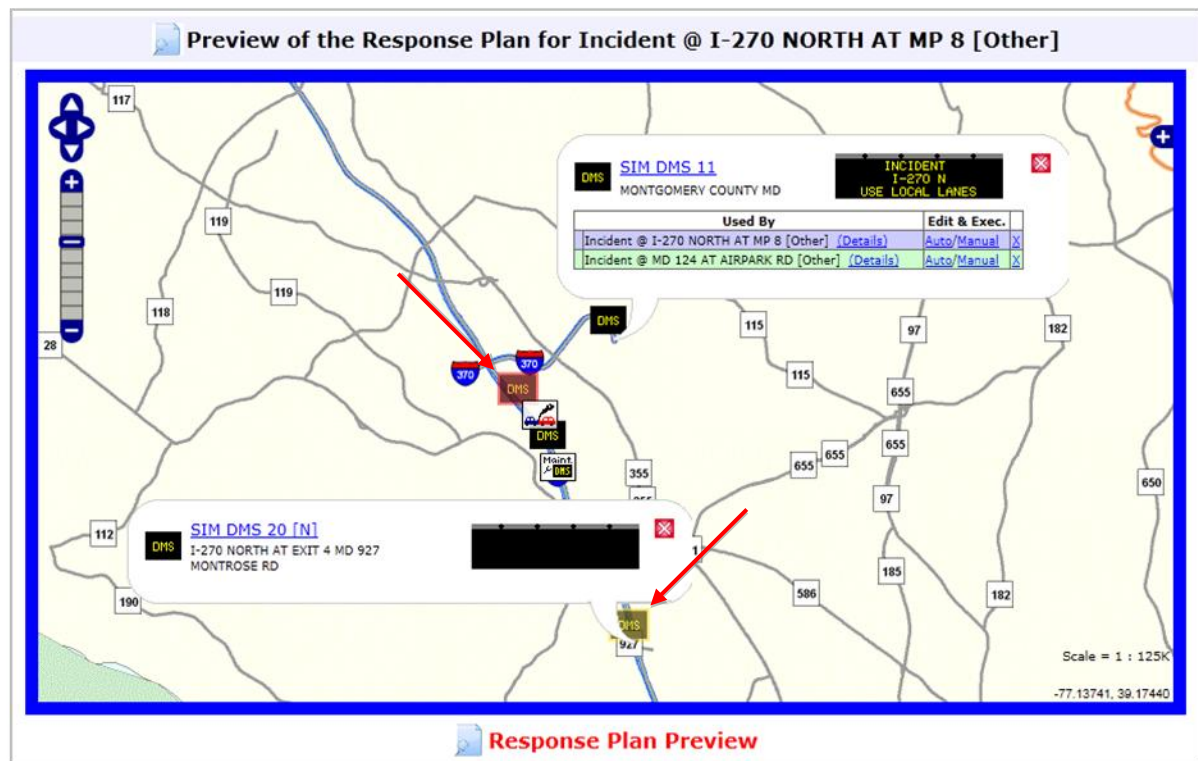
### 5.1.1.15 Response Plan Preview Map

A user may view a preview of the current response plan on the map by clicking the Preview on Map button (see Figure 5-20 below) at the bottom of the response plan section on the traffic event details page.



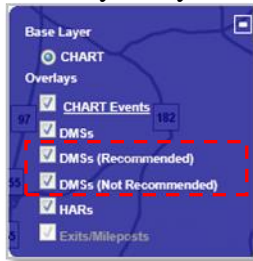
**Figure 5-20. The Button for Viewing a Preview of the Response Plan on a Map.**

The preview map will contain all the DMSs in the response plan, as well as the additional suggested DMSs that are not currently in the response plan (see Figure 5-21 below). The additional suggested DMSs will be highlighted in yellow on the map. Any DMSs in the response plan that are not recommended will be highlighted in red on the map. Note: A legend indicating that devices with red highlighting are not recommended and devices with yellow highlighting are recommended will be added during implementation. The callouts for the DMSs in the response plan will show the current message (or blank if no message has been configured). The callouts for the additional suggested DMSs will show a blank message.



**Figure 5-21. The Response Plan Preview Map Showing Recommended and Not Recommended DMSs.**

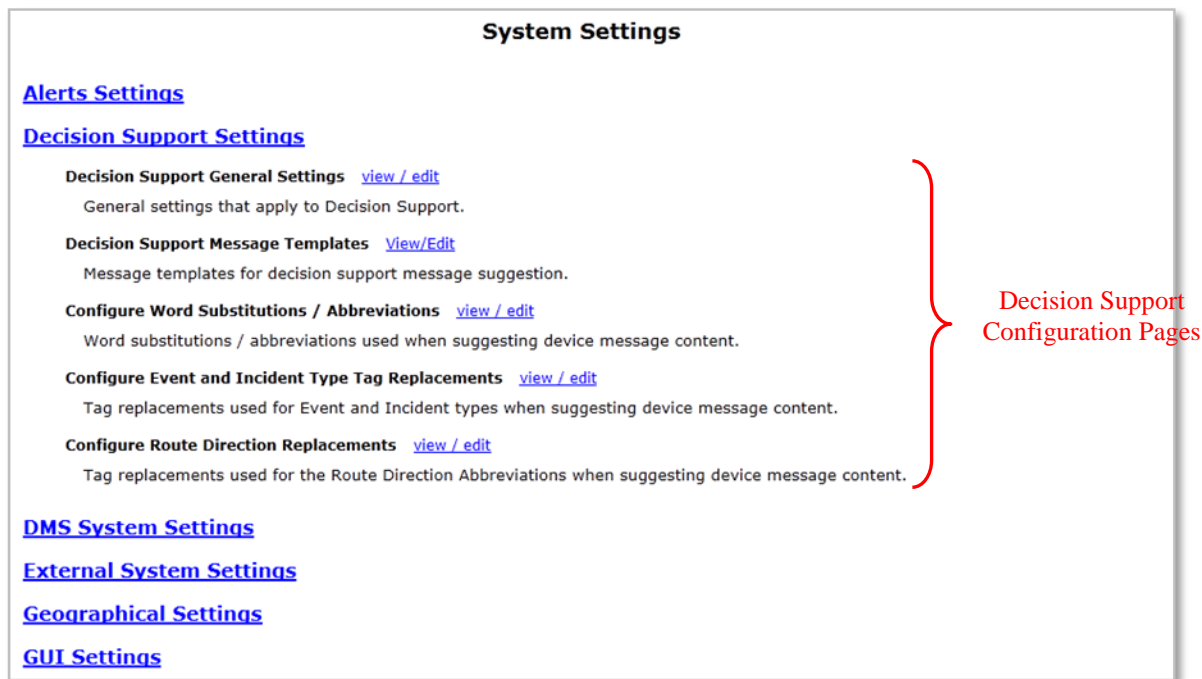
The DMSs on the Response Plan Preview map will be shown on 3 layers (see Figure 5-22 below). The DMSs layer will contain DMSs that are currently in the response plan and are not in the not recommended group. The DMSs (Recommended) layer will contain additional DMSs with a proximity of upstream that are not currently in the response plan but are recommended. The DMSs (Not Recommended) layer will contain DMSs with a proximity of downstream that are currently in the response plan but are not recommended. A user may show or hide any of these layers by clicking on the checkbox next to the layer name.



**Figure 5-22. The Response Plan Preview Map Layer Switcher Showing Recommended and Not Recommended DMS Layers.**

### 5.1.2 Configuring Decision Support Settings

The criteria for suggesting DMSs includes: the type of traffic event, the distance of a DMS from the traffic event, the proximity of a DMS to the traffic event, the geometry of the DMS, the maximum number of pages for the DMS, and the tags in the message template. Factors relating to each of these criteria are configurable (as mentioned in section 5.1.1.1). From the System Setting page (in the System Profile), a user may view any of the Decision Support configuration pages (see Figure 5-23 below). Note: All decision support settings headers will be updated during implementation to indicate they are DMS specific.



**Figure 5-23. The Configuring Decision Support Settings Page.**

### 5.1.2.1 General Settings

The Decision Support General Setting page contains configuration settings for distance, proximity type, and route type. These settings are used when determining if a DMS or DMS Message Template should be suggested for a traffic event. A user can edit the distance, proximity type, or route type values (either by entering a new number value or clicking on a checkbox) and save the form by clicking on the Save Changes button.

### 5.1.2.2 Configuring Distances

When a user requests DMS suggestions, the method of determining whether a DMS should be suggested is partly based on the distance of the DMS from the traffic event and the percentage of lane closures for the traffic event. A user may configure the 3 maximum distances (in miles) - one for each of the 3 distance types: Immediate, Near, and Far (see Figure 5-24 below) - by entering a new value in the textbox. A user may also configure a lane closure percentage for each of the 3 distance types by entering a new value in the textbox. These settings are used in the following way to determine whether a DMS should be suggested for a traffic event. First, the percentage of lane closures for the traffic event is determined. Next, the furthest applicable distance type is determined based on the percentage of lane closures for the traffic event and the configured lane closures percentage. If the distance from the DMS to the traffic event is less than the configured maximum distance for the distance type, the DMS is considered a valid suggestion.

Distance Type	Max Distance	Lanes Closed
Immediate	2.0 (miles)	0 (%)
Near	6.0 (miles)	0 (%)
Far	12.0 (miles)	25 (%)

**Figure 5-24. The Distances Settings Form Showing the Maximum Distances and the Percentage of Lanes Closed.**

### 5.1.2.3 Configuring Proximity Types

When a user requests DMS suggestions, the method of determining whether a DMS should be suggested is partly based on the proximity of the DMS to the traffic event. A user may configure the applicable proximity types by clicking on the checkbox beside each applicable proximity type name (see Figure 5-25 below). If a checkbox is left unchecked, no DMSs with that proximity type will be suggested for a traffic event.

**Proximity Types to Suggest Messages For**

- ☒ Upstream
- ☒ Same Route
- ☒ Same Route, Same Dir
- ☒ Same Route, Opposite Dir
- ☐ Downstream
- ☐ Opposite Dir, Upstream
- ☒ Opposite Dir, Downstream
- ☒ Other Route, Possibly Drivable
- ☒ Other Route, Drivable
- ☐ No Direct Route
- ☐ Undetermined

**Figure 5-25. The Proximity Settings Showing the Proximities for Which to Suggest Messages.**

*Note: User feedback indicated that the above list of proximities was confusing. For that reason, the final implementation of this feature will include only 3 options (or checkboxes): a) “Same route only?”, b) “Same direction only?”, and c) “Upstream only?”. The same direction and upstream checkboxes will only be available if the same route checkbox is checked.*

### 5.1.2.4 Configuring Route Types


When a user requests DMS suggestions, the method of determining whether a DMS Message Template should be suggested for a DMS is partly based on the route type of the traffic event and the route tags (Route Type, Route Number, and Route Name) in the template message. A user may configure the applicable route types by clicking on the checkbox beside each route type name. For each route type, the Route Type / Number and Route Name may be configured separately. If the Route Type / Number checkbox is not checked for the route type of a traffic event and either a Route Type or Route Number tag exists in a DMS message template, that template will not be suggested for that DMS. Likewise, if the Route Name checkbox is not checked for the route type of a traffic event and a Route Name tag exists in a DMS message template, that template will not be suggested for that DMS.

Route Types to Suggest Messages For		
	Suggest Messages for Templates That Contain	
Interstate	<input checked="" type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
State	<input checked="" type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
US Route	<input checked="" type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
County	<input type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
US Government	<input type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
Municipal	<input type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
Other Public	<input type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
Other State Road	<input type="checkbox"/> Route Type / Number	<input checked="" type="checkbox"/> Route Name
Other	<input type="checkbox"/> Route Type / Number	<input type="checkbox"/> Route Name
Unknown	<input type="checkbox"/> Route Type / Number	<input type="checkbox"/> Route Name

**Figure 5-26. The Route Types Settings Showing the Route Types for Which to Suggest Messages.**

### 5.1.3 Configuring DMS Message Templates

Message suggestions for a DMS are based on a configured list of DMS Message Templates (see Figure 5-27 below). Each template contains a message, a list of applicable event types, a list of applicable proximities, a list of applicable distances, and a list of applicable geometries.

Decision Support DMS Templates (7) <a href="#">Set Columns</a>						
<a href="#">Add Template</a> 						
Message Template ▾	Event Type	Proximity	Distance	Geometry	Example Message	Actions
--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾		
[PT2500][JL3]<EVENT_TYPE> [NL][JL3] <ROUTE_TYPE>-<ROUTE_NUM> <ROUTE_DIR>[NL][JL3]BE ALERT[NL][JL3]	Incident	Upstream	Immediate	3 X 8	ACDT I-270 NB BE ALERT	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL3]<EVENT_TYPE> [NL][JL3] <ROUTE_TYPE>-<ROUTE_NUM> <ROUTE_DIR>[NL][JL3] <LANE_CLOSURES>[NL][JL3]	Congestion Event Disabled Vehicle Event Incident	Same Route, Same Dir Upstream Same Route	Far Near	3 X 18 3 X 21 4 X 24	ACDT I-270 NB LFT LANE CLOSED	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL3]<EVENT_TYPE> [NL][JL3]<ROUTE_NAME>[NL][JL3] <LANE_CLOSURES>[NL][JL3]	Incident	Upstream Opposite Dir, Downstream	Immediate Far Near	3 X 18 3 X 21 4 X 24	ACDT EISENHOWER MEM HWY LFT LANE CLOSED	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL3]<EVENT_TYPE> [NL][JL3]<ROUTE_NAME> <ROUTE_DIR>[NL][JL3] <LANE_CLOSURES>[NL][JL3] [NP][PT2500][JL3][NL][JL3]USE CAUTION[NL][JL3][NL][JL3]	Disabled Vehicle Event Incident	Same Route, Same Dir Upstream Other Route Opposite Dir, Downstream Other Route Same Route Same Route, Opposite Dir	Immediate Far Near	3 X 21	USE CAUTION	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL3]<EVENT_TYPE> AHEAD[NL] [JL3]<ROUTE_TYPE>-<ROUTE_NUM> <ROUTE_DIR>[NL][JL3]STAY ALERT[NL][JL3]	Disabled Vehicle Event Incident	Upstream	Immediate Far Near	3 X 18 3 X 21	ACDT AHEAD I-270 NB STAY ALERT	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL3]<EVENT_TYPE> AHEAD[NL][JL3]<ROUTE_NAME> [NL][JL3]<LANE_CLOSURES>	Incident Planned Closure	Upstream	Immediate Far Near	3 X 18 3 X 21 4 X 24	ACDT AHEAD EISENHOWER MEM HWY LFT LANE CLOSED	<a href="#">Edit</a> <a href="#">Remove</a>
[PT2500][JL2]<EVENT_TYPE> [NL][JL3]<ROUTE_NAME> <ROUTE_DIR>[NL][JL4]USE CAUTION[NL][JL3]	Disabled Vehicle Event Incident	Upstream Opposite Dir, Downstream	Immediate Far Near	3 X 21 4 X 24	ACDT EISENHOWER MEM HWY NB USE CAUTION	<a href="#">Edit</a> <a href="#">Remove</a>
<a href="#">Add Template</a>						

**Figure 5-27. The Decision Support DMS Message Template List Page.**

### 5.1.3.1 Adding a New DMS Message Template


A user can create a new template by clicking on the Add Template link on the DMS message template list page which will open the template editor. The user then must configure the message, applicable event types, applicable proximities, applicable distances, and applicable geometries. Once all values have been entered, the user can click on the OK button to save the template.

#### Template Message

The message is composed of tags and user entered text. The available tags include: Event Type, Incident Type, Route Type, Route Number, Route Name, Route Direction, Proximity, Exit Number, Exit Road Name, and Lane Closures. When a message is edited, the user is shown a preview of the new message text (see Figure 5-28 below). When a suggested message is created using a message template, the tags are replaced with values from the traffic event. If there is no data from the traffic event for one or more of the tags in the message template, the suggested message will not be created.



**Add Decision Support Message Template**



**Page 1**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<EVENT_TYPE> <a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<INT_EXIT_PROXIMITY> EXIT <a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>

**Traffic Event Types:**

- Action Event
- Congestion Event
- Disabled Vehicle Event
- Incident
- Planned Closure
- Safety Message Event
- Special Event
- Weather Service Event

**Proximities:**

- Upstream
- Same Route
- Same Route, Same Dir
- Same Route, Opposite Dir
- Downstream
- Opposite Dir, Upstream
- Opposite Dir, Downstream
- Other Route
- Other Route
- No Direct Route
- Out Of Range
- Undetermined

**Page 2**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>

**Event**

- [Event Type](#)
- [Incident Type](#)

**Route**

- [Route Type](#)
- [Route Number](#)
- [Route Name](#)
- [Route Direction](#)

**Exit**

- [Proximity](#)
- [Exit Number](#)
- [Exit Road Name](#)

**Lane Closure**

- [Lane Closures](#)

**Geometries:**

- 3 X 8
- 3 X 18**
- 3 X 21
- 4 X 24

**Beacons:** ☐ **Page On Time (Seconds):** 2.5 **Page Off Time (Seconds):** 0.0

[OK](#)
[Cancel](#)
[Check Spelling](#)
[Show Def. Editor](#)

**Figure 5-28. The DMS Message Template Editor with the Tag Selector Popup Displayed.**

#### Applicable Traffic Event Types

A user must select one more traffic event types that are applicable to this template (see Figure 5-29 below). Suggested messages created using a message template will only be created for events that match one of the applicable traffic event types.

#### Applicable Proximities

A user must select one or more proximities that are applicable to this template (see Figure 5-29 below). Suggested messages created using a message template will only be created when the proximity of the DMS to the traffic event matches one or more of the applicable proximities.



*Note: User feedback indicated that the above list of proximities was confusing. For that reason, the final implementation of this feature will include only 3 options (or checkboxes): a) “Same route only?”, b) “Same direction only?”, and c) “Upstream only?”. The same direction and upstream checkboxes will only be available if the same route checkbox is checked.*

### Applicable Distances

A user must select one or more distances that are applicable to this template (see Figure 5-29 below). Suggested messages created using a message template will only be created when the distance of the DMS from the traffic event is less than one of the configured maximum distances for one of the applicable distances.

**Add Decision Support Message Template**

ACDT

I-270 NB

LFT LNE CLOSED

**Page 1**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<EVENT_TYPE> <a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<ROUTE_TYPE>-<ROUTE_NUM> < <a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<LANE_CLOSURES> <a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>

**Page 2**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<a href="#">Tag</a>

**Traffic Event Types:**

- Action Event
- Congestion Event
- Disabled Vehicle Event
- Incident
- Planned Closure
- Safety Message Event
- Special Event
- Weather Service Event

**Proximities:**

- Upstream
- Same Route
- Same Route, Same Dir
- Same Route, Opposite Dir
- Downstream
- Opposite Dir, Upstream
- Opposite Dir, Downstream
- Other Route
- Other Route
- No Direct Route
- Out Of Range
- Undetermined

**Distances:**

- Immediate
- Near
- Far

**Geometries:**

- 3 X 8
- 3 X 18
- 3 X 21
- 4 X 24

**Beacons:** ☐ **Page On Time (Seconds):** 2.5 **Page Off Time (Seconds):** 0.0

**Figure 5-29. The DMS Message Template Editor With All Values Set.**

#### Applicable Geometries

A user must select one or more geometries that are applicable to this template (see Figure 5-29 above). The system will display a warning (see Figure 5-30 below) if the sample message cannot be displayed on any of the selected geometries. Suggested messages created using a message template will only be created when the geometry of the DMS matches one or more of the applicable geometries and the message will fit on the DMS.

**Edit Decision Support Message Template**

Could not display image, no filename specified.

**Page 1**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text" value="&lt;EVENT_TYPE&gt;"/> <a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text" value="&lt;ROUTE_TYPE&gt;-&lt;ROUTE_NUM&gt; &lt;"/> <a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text" value="&lt;LANE_CLOSURES&gt;"/> <a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text"/> <a href="#">Tag</a>

**Page 2**

Row 1:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text"/> <a href="#">Tag</a>
Row 2:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text"/> <a href="#">Tag</a>
Row 3:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text"/> <a href="#">Tag</a>
Row 4:	<input type="radio"/> Left <input checked="" type="radio"/> Center <input type="radio"/> Right	<input type="text"/> <a href="#">Tag</a>

LINE TOO LONG FOR THE SELECTED GEOMETRY

**Traffic Event Types:**

- Action Event
- Congestion Event
- Disabled Vehicle Event
- Incident
- Planned Closure
- Safety Message Event
- Special Event
- Weather Service Event

**Proximities:**

- Upstream
- Same Route
- Same Route, Same Dir
- Same Route, Opposite Dir
- Downstream
- Opposite Dir, Upstream
- Opposite Dir, Downstream
- Other Route
- Other Route
- No Direct Route
- Out Of Range
- Undetermined

**Distances:**

- Immediate
- Near
- Far

**Geometries:**

- 3 X 8
- 3 X 18
- 3 X 21
- 4 X 24

**Beacons:** ☐
**Page On Time (Seconds):** 
**Page Off Time (Seconds):**

**Figure 5-30. DMS Message Template Editor With Warning About Message Too Long for Geometry.**

#### 5.1.4 Configuring Word Substitution/Abbreviation List

A user may configure word substitutions (or abbreviations) that are used when generating a suggested message from a message template (see Figure 5-31 below). The substitutions are used to replace values from the traffic event that are replacing tags in the message template. The substitutions are not used to replace text entered by the user. First, the substitution is performed

using the long version for each word. If the suggested message will not fit on the DMS, the substitution is then performed using the short version for each word.

**Decision Support Word Substitution / Abbreviations List**

[Add Substitution / Abbreviation](#) ←

Word	Short Version	Long Version	Actions
LANE	LN	LNE	<a href="#">Edit</a> <a href="#">Remove</a>
LEFT	L	LFT	<a href="#">Edit</a> <a href="#">Remove</a>
RIGHT	R	RT	<a href="#">Edit</a> <a href="#">Remove</a>
ROAD	R	RD	<a href="#">Edit</a> <a href="#">Remove</a>

[Add Substitution / Abbreviation](#)

**Figure 5-31. The Decision Support Word Substitution / Abbreviation List.**

A user may configure a new word substitution by clicking on the Add Substitution / Abbreviation link on the Word Substitution / Abbreviations list page. This will open the Add Word Substitution / Abbreviation page (see Figure 5-32 below). A user can enter a Word, Short Version, and Long Version and save the substitution by clicking on the Add Substitution / Abbreviation button. A user can enter a blank for the Short Version or Long Version substitution in order to remove the word from the message suggestion. It is valid for the user to enter blanks (not text) for short version, long version or both. If both the long and short versions are blank the word or phrase will be removed from the message completely.

**Add Word Substitution / Abbreviation (Decision Support)**

**Word Substitution / Abbreviation**

Word: STREET

Short Version: ST

Long Version: STR

[Add Substitution / Abbreviation](#) [Cancel](#)

**Figure 5-32. Adding a New Word Substitution / Abbreviation for Decision Support.**

### 5.1.5 Configure Event and Incident Type Tag Replacements

A user may configure traffic event type and incident type word replacements that are used when generating a suggested message from a message template (see Figure 5-33 below). The replacements are used to replace event type and incident type values from the traffic event that are replacing event type and incident type tags in the message template. First, the replacement is performed using the long version for each word. If the suggested message will not fit on the DMS, the replacement is then performed using the short version for each word. A user can configure a new replacement word on the Event and Incident Type Tag Replacements page by simply editing a replacement word and saving the form by clicking on the Save Changes button.

**Decision Support Event and Incident Type Tag Replacements**

Traffic Event Type Tag Replacements

Event Type	Short Version	Long Version
Action Event	INCIDENT	INCIDENT
Congestion Event	CONGESTION	CONGESTION
Disabled Vehicle Event	DISABLED VEHICLE	DISABLED VEHICLE
Incident	INCIDENT	INCIDENT
Planned Closure	ROADWAY CLOSURE	ROADWAY CLOSURE
Safety Message Event	STAY ALERT	STAY ALERT
Special Event	SPECIAL EVENT	SPECIAL EVENT
Weather Service Event	INCLEMENT WEATHER	INCLEMENT WEATHER

Incident Type Tag Replacements

Incident Type	Short Version	Long Version
Collision, Fatality	ACCIDENT	ACCIDENT
Collision, Personal Injury	ACCIDENT	ACCIDENT
Collision, Property Damage	ACCIDENT	ACCIDENT
Debris In Roadway	ACCIDENT	ACCIDENT
Disabled In Roadway	ACCIDENT	ACCIDENT
Emergency Roadwork	ACCIDENT	ACCIDENT
Off Road Activity	ACCIDENT	ACCIDENT
Other	ACCIDENT	ACCIDENT
Police Activity	ACCIDENT	ACCIDENT
Utility Problem	ACCIDENT	ACCIDENT
Vehicle Fire	ACCIDENT	ACCIDENT
Weather Closure	ACCIDENT	ACCIDENT
Weather Closure, Debris	ACCIDENT	ACCIDENT
Weather Closure, High Water	ACCIDENT	ACCIDENT
Weather Closure, Winter Precip.	ACCIDENT	ACCIDENT

Save Changes

Cancel

**Figure 5-33. Configuring Replacement Text for the Traffic Event Type and Incident Type Tags.**

### 5.1.6 Configure Route Direction Replacements

A user may configure route direction and additional direction replacements that are used when generating a suggested message from a message template (see Figure 5-34 below). The replacements are used to replace directions from the traffic event that are replacing tags in the message template. A user can configure a new replacement word on the Route Direction Abbreviation Replacements page by simply editing a replacement word and saving the form by clicking on the Save Changes button. A user can enter a blank replacement for a route direction to indicate that the route direction should not be used in message content.

### Decision Support Route Direction Abbreviation Replacements

#### Valid Roadway Direction Abbreviation Replacements

None	NONE
North	N
East	E
South	S
West	W
Inner Loop	IL
Outer Loop	OL
South/North	S/N
East/West	E/W
Inner Loop/Outer Loop	IL/OL





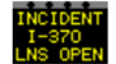
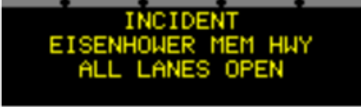


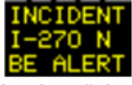

#### Additional Direction Replacements

Southeast	SE
Southwest	SW
Northeast	NE
Northwest	NW
Other	OTHER
Other (no info)	OTHER

**Figure 5-34. Configuring Replacement Text for the Valid Roadway Directions and Additional Directions.**

### 5.1.7 Additional Feature in Response Plan

A preview of the current message for each DMS in the response plan is available by hovering the mouse pointer over the name of the DMS (see below). If the DMS does not have a current message, the message will be blank.

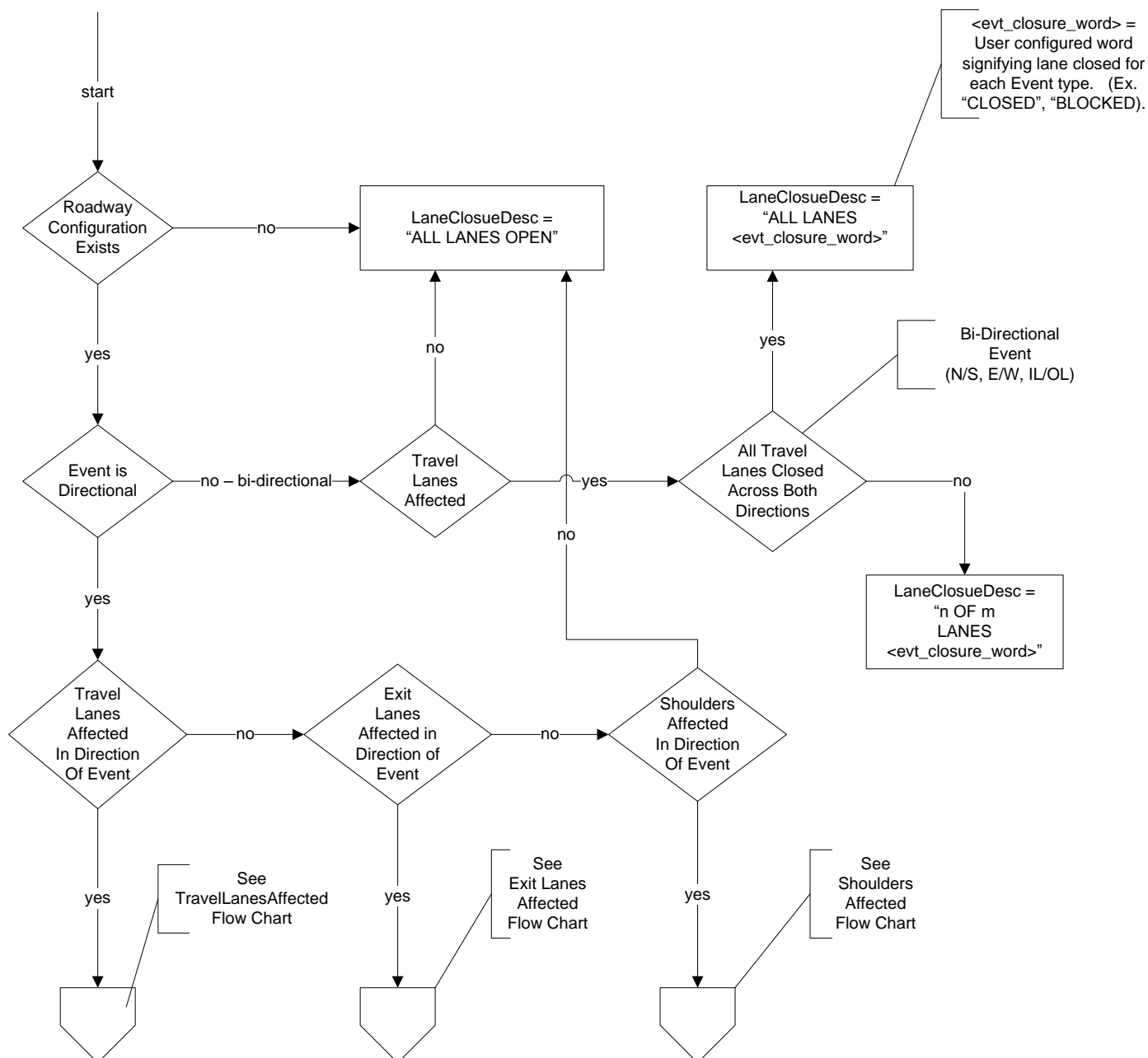
Target Device	Proposed Message	Status
 DMS SIM 123 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>
 SIM 1 <b>Current Message</b>  <a href="#">Device Details</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>
 SIM DMS 8 <a href="#">Device Details</a> / <a href="#">Device Queue</a>	 <a href="#">Edit (Auto)</a> <a href="#">Edit (Manual)</a>	 <a href="#">Execute</a> <a href="#">Revoke Execution</a> <a href="#">Remove</a>

**Figure 5-35. DMS Current Message Preview on the Traffic Event Details Page.**

## 5.2 Lane Closure Description Generation Flow Chart

The following set of flow charts describe the logic used to create the Lane Closure Description string used for CHART Decision Support Message Suggestion functionality. The logic is described in requirement 4.2.3.2.1.2.10.1.1.4.3 and its children.

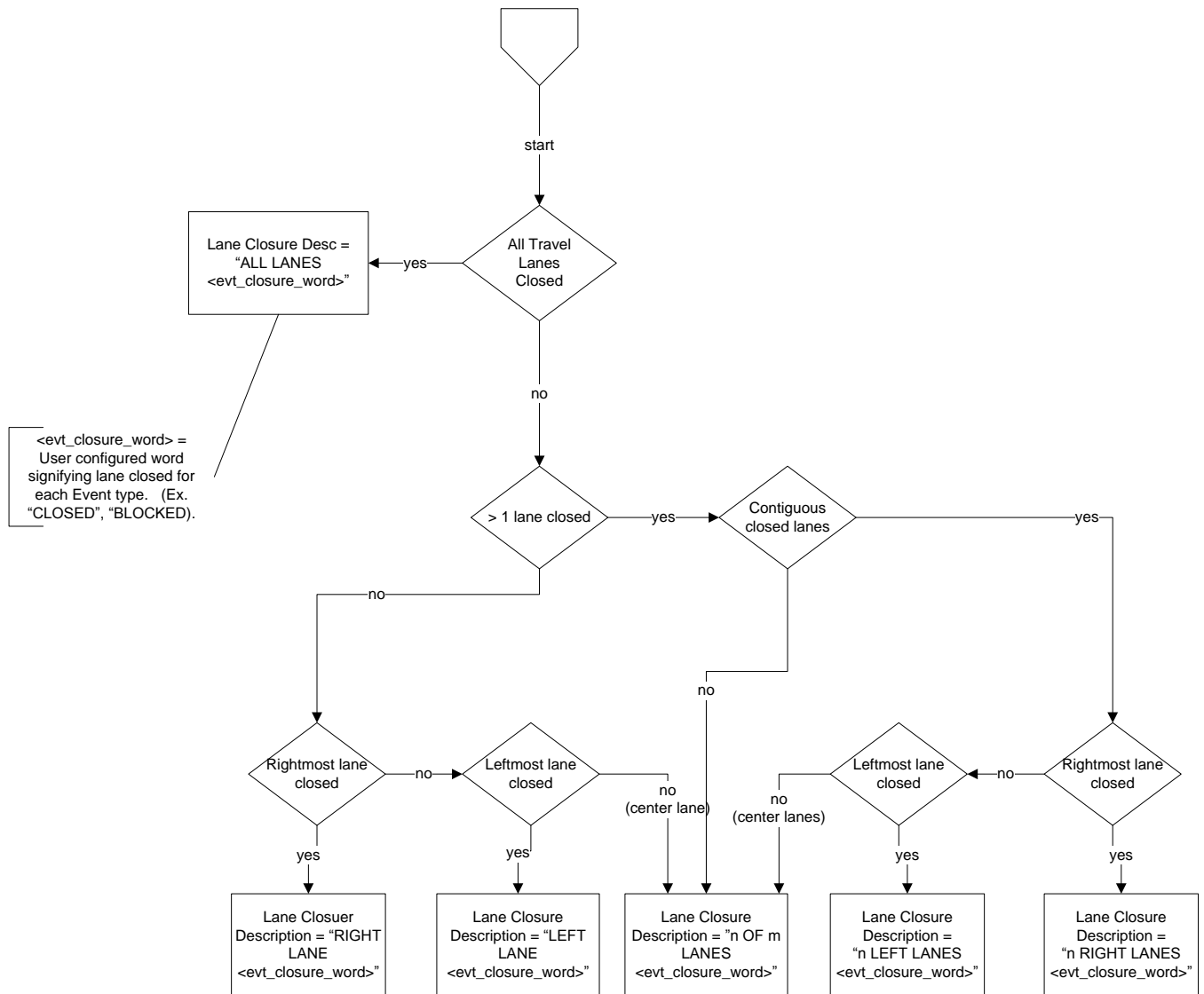
### Lane Closure Description Main Logic:



**Figure 5-36 Lane Closure Description Main Logic**



## Travel Lanes Affected Logic:



**Figure 5-37 Travel Lanes Affected Logic**

## Exits Lanes Affected Logic:

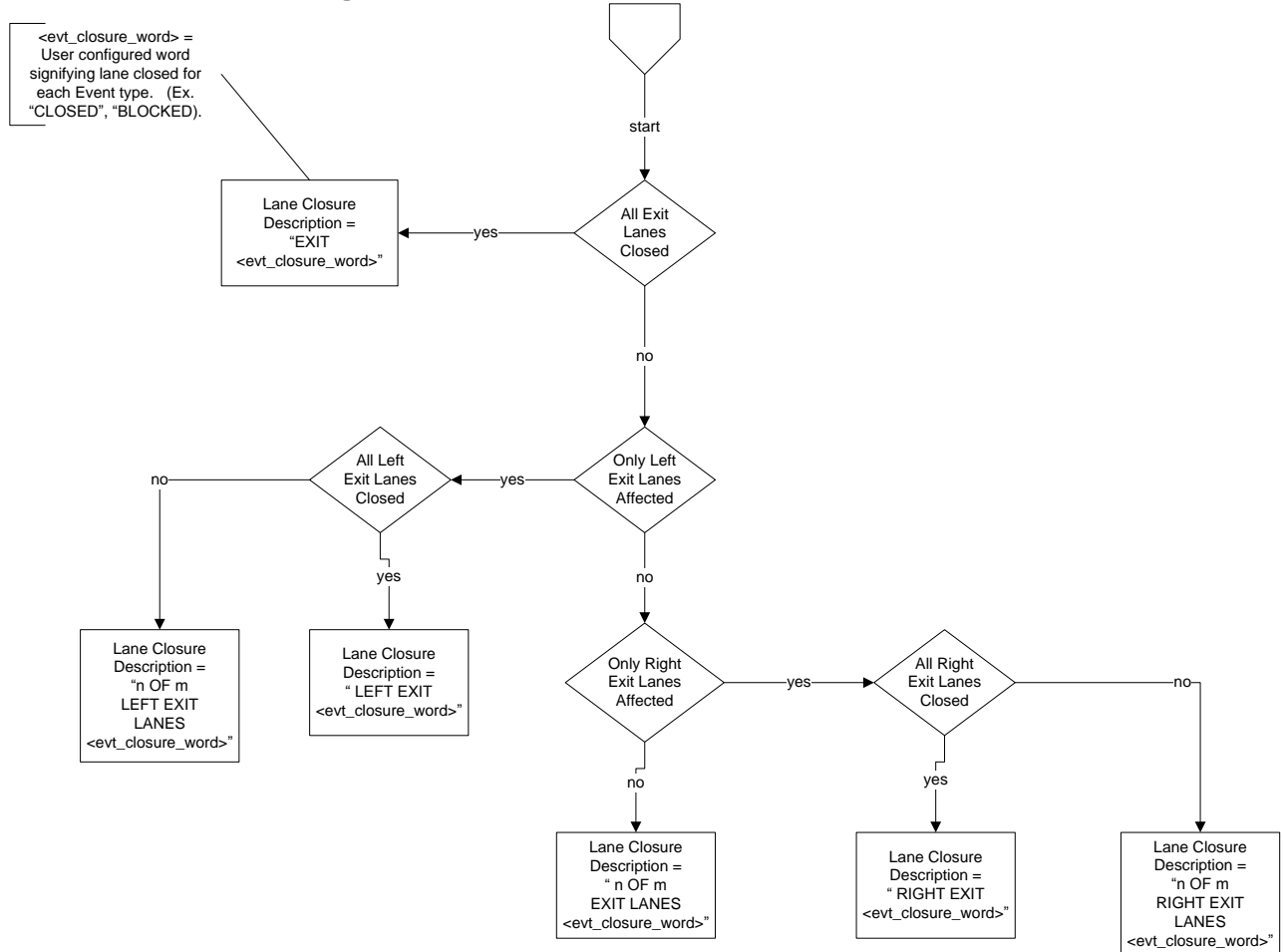
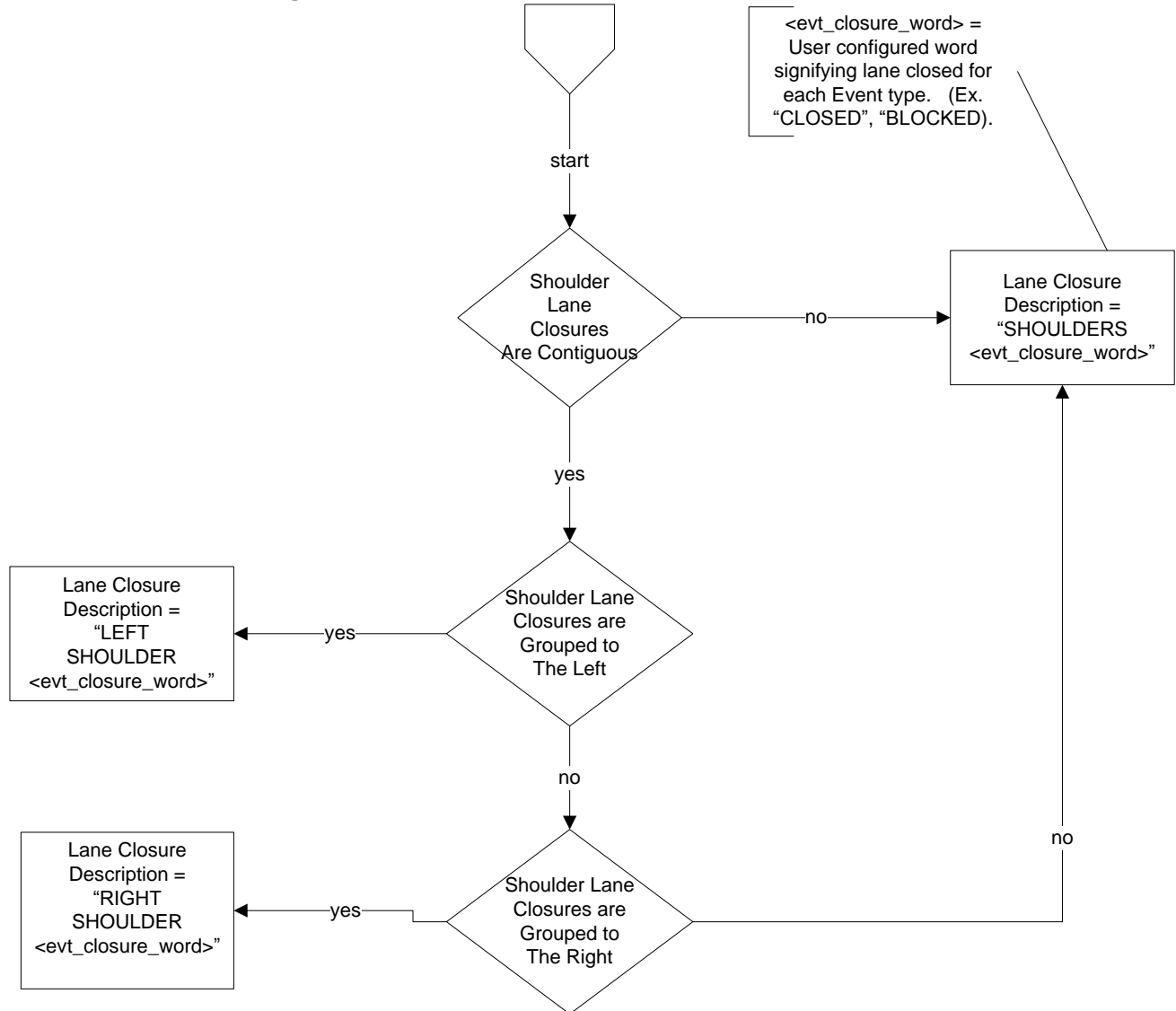


Figure 5-38 Exits Lanes Affected Logic

### Shoulders Affected Logic:



**Figure 5-39 Shoulders Affected Logic**

### 5.3.1.1 Common (Class Diagram)

```

classDiagram
    class UniqueIdentifier {
        <<interface>>
        long
        getID() long
    }
    class GeoLocation {
        <<interface>>
        string
        getLocationDesc() string
        getLocation() GeoLocation
        getLocation() GeoLocation
    }
    class Service {
        <<interface>>
        string
        void
        getIName() string
        getIName() ApplicationVersion
        getIName() ApplicationVersion
        shutdown() AccessToken
    }
    class UserName {
        <<type>>
        string
    }
    class Password {
        <<type>>
        string
    }
    class Direction {
        <<interface>>
        Direction
        const short OTHER_NO_ADDITIONAL_INFO
        const short OTHER_ADDITIONAL_INFO
        const short NORTH
        const short NORTH_EAST
        const short SOUTH_EAST
        const short SOUTH
        const short SOUTH_WEST
        const short WEST
        const short NORTH_WEST
        const short INNER_LOOP
        const short OUTER_LOOP
    }
    class Timestamp {
        <<type>>
        long
        timestamp
    }
    class NetworkConnectionSite {
        <<type>>
        string
        void
        getIName() string
        getIName() ApplicationVersion
        getIName() ApplicationVersion
        shutdown() AccessToken
    }
    class TrafficParameters {
        <<struct>>
        int m_speedData
        int m_volumeData
        int m_percentOccupancy
        SpeedRange m_speedRange
    }
    class Direction {
        <<type>>
        short
    }
    class Source {
        <<type>>
        string
        string
        string
        string
    }
    class CHARTZException {
        <<exception>>
        string reason
    }
    class UnsupperAdapterException {
        <<exception>>
        string reason
    }
    class ComponentVersion {
        <<type>>
        string name
        string version
    }
    class TimeSpecificationType {
        <<enumration>>
        TIME_ABSOLUTE
        TIME_RELATIVE
    }
    class AccessDenied {
        <<exception>>
        string reason
        string requiresRights
    }
    class SpecifiedObjectNotFound {
        <<exception>>
        string reason
    }
    class InvalidState {
        <<exception>>
        string reason
    }
    class ApplicationVersion {
        <<type>>
        string applicationName
        ComponentVersionList componentVersions
    }
    class AbsoluteOrRelativeTime {
        <<enumration>>
        AbsoluteOrRelativeTime
        discriminator TimeSpecificationType
        timestamp absoluteTime
        timestamp relativeTime
    }
    class DuplicateData {
        <<exception>>
        string reason
    }
    class CommandStatus {
        <<interface>>
        update() string
        void
        completed() boolean
        commandsSuccessful()
        completedSameStatus() boolean
        commandsSuccessful()
    }
    class EventChannelPortStatus {
        <<type>>
        string
    }
    class ConnectFailure {
        <<exception>>
        string reason
    }
    class PortOpenFailure {
        <<exception>>
        string reason
    }
    class PortStatus {
        <<enumration>>
        STATUS_OK
        STATUS_MARGINAL
        STATUS_FAILED
        STATUS_DISABLEDFuture
    }
    class DataOrIOException {
        <<exception>>
        string reason
    }
    class PortType {
        <<enumration>>
        ISDN
        MODEM
        POIS
        MODEM
        DIRECT
        RS232
        TELEPHONE
        TCP/IP
    }
    class Priority {
        <<enumration>>
        PRIORITY_POLLING
        PRIORITY_OK
        DEMAND
    }
    class PortStatusInfo {
        <<type>>
        identifier id
        string name
        PortType type
        PortStatus status
    }
    class ProximityDirectionType {
        <<enumration>>
        SAME_DIR
        OPPOSITE_DIR
        UNKNOWN_DIR
    }
    class ProximityDistance {
        <<type>>
        DistanceType
        distanceType
        PortType type
        PortStatus status
    }
    class ProximityInfo {
        <<type>>
        boolean sameRate
        ProximityDirectionType directionType
        boolean upstream
    }
    class ProximityRate {
        <<type>>
        boolean sameRate
        ProximityDirectionType directionType
        boolean upstream
    }
    class PortStatusChangeEventInfo {
        <<type>>
        PortStatusInfo info
    }
    class PortStatusInfo {
        <<type>>
        PortStatusInfo info
    }
    class AllThesePortRelatedClassesAreMovedFromR333ToR333 {
        <<note>>
        All these port related classes are moved from R333 to R333
    }
    UniqueIdentifier --> GeoLocation
    GeoLocation --> Service
    Service --> UserName
    UserName --> Password
    Password --> Direction
    Direction --> Timestamp
    Timestamp --> NetworkConnectionSite
    NetworkConnectionSite --> TrafficParameters
    TrafficParameters --> Direction
    Direction --> Source
    Source --> CHARTZException
    CHARTZException --> UnsupperAdapterException
    UnsupperAdapterException --> ComponentVersion
    ComponentVersion --> TimeSpecificationType
    TimeSpecificationType --> AccessDenied
    AccessDenied --> SpecifiedObjectNotFound
    SpecifiedObjectNotFound --> InvalidState
    InvalidState --> ApplicationVersion
    ApplicationVersion --> AbsoluteOrRelativeTime
    AbsoluteOrRelativeTime --> DuplicateData
    DuplicateData --> CommandStatus
    CommandStatus --> EventChannelPortStatus
    EventChannelPortStatus --> ConnectFailure
    ConnectFailure --> PortOpenFailure
    PortOpenFailure --> PortStatus
    PortStatus --> DataOrIOException
    DataOrIOException --> PortType
    PortType --> Priority
    Priority --> PortStatusInfo
    PortStatusInfo --> ProximityDirectionType
    ProximityDirectionType --> ProximityDistance
    ProximityDistance --> ProximityInfo
    ProximityInfo --> ProximityRate
    ProximityRate --> PortStatusChangeEventInfo
    PortStatusChangeEventInfo --> PortStatusInfo
    PortStatusInfo --> AllThesePortRelatedClassesAreMovedFromR333ToR333
  
```

**Figure 5-40. Common (Class Diagram)**

This class represents an access denied, or "no rights" failure.

#### **5.3.1.1.3 ApplicationVersion (Class)**

This structure contains the name of the application and information about the versions of its components.

#### **5.3.1.1.4 CHART2Exception (Class)**

Generic exception class for the CHART2 system. This class can be used for throwing very generic exceptions which require no special processing by the client. It supports a reason string which may be shown to any user and a debug string which will contain detailed information useful in determining the cause of the problem.

#### **5.3.1.1.5 CommandStatus (Class)**

The CommandStatus CORBA interface is used to allow a calling process to be notified of the progress of a long-running asynchronous operation. This is normally used when field communications are involved to complete a method call. The most common use is to allow a GUI to show the user the progress of an operation. It can also be used and watched by a server process when it needs to call on another server process to complete an operation. The long running operation typically calls back to the CommandStatus object periodically as the command is being executed, to provide in-progress status information, and it always makes a final call to the CommandStatus when the operation has completed. The final call to the CommandStatus from the long running operation indicates the success or failure of the command.

#### **5.3.1.1.6 ComponentVersion (Class)**

This structure contains the name and version number of the software component.

#### **5.3.1.1.7 ConnectFailure (Class)**

This exception is a catch-all for exceptions that do not fit in a more specific exception that can be thrown during a connection attempt.

#### **5.3.1.1.8 DataPortIOException (Class)**

This exception is used to indicate an Input/Output error has occurred.

#### **5.3.1.1.9 Direction (Class)**

This type defines a short value that is used to indicate a direction of travel as defined in DirectionValues.

#### **5.3.1.1.10 DirectionValues (Class)**

This interface contains constants for directions as defined in the TMDD.

#### **5.3.1.1.11 DistanceType (Class)**

This enumeration represents different values for types of distance stated in miles (roadway miles, straight line miles).

#### **5.3.1.1.12 DuplicateData (Class)**

This exception is thrown when an object is to be added to the system, but the system already contains an object with equivalent data.

#### **5.3.1.1.13 EVENT\_CHANNEL\_PORT\_STATUS (Class)**

This is a static string that contains the name of the event channel used to push events relating to the change in Port status. The following PortEventTypes are pushed on EVENT\_CHANNEL\_PORT\_STATUS channel: PortStatusChanged

#### **5.3.1.1.14 GeoLocatable (Class)**

This interface is implemented by objects that can provide location information to their users.

#### **5.3.1.1.15 GetPortTimeout (Class)**

This class is an exception that is thrown by a PortManager when a request to acquire a port of a given type cannot be fulfilled within the timeout specified.

#### **5.3.1.1.16 InvalidState (Class)**

This exception is thrown when an operation is attempted on an object that is not in a valid state to perform the operation.

#### **5.3.1.1.17 NetworkConnectionSite (Class)**

The NetworkConnectionSite class contains a string that is used to specify where a service is running. This field is useful for administrators in debugging problems should an object become "software comm failed".. It is included in the Chart2DMSSStatus.

#### **5.3.1.1.18 Password (Class)**

Typedef used to define the type of a Password.

#### **5.3.1.1.19 PortEventType (Class)**

This enum defines the types of CORBA events that are pushed on a Field Communications event channel.

#### **5.3.1.1.20 PortOpenFailure (Class)**

This exception is thrown if there is an error opening the port while attempting a connection.

This exception would most likely only occur if there is another application accessing the physical com port, which would be true if debugging activities were being done on a port while the FieldCommunications service is still running.

#### **5.3.1.1.21 PortStatus (Class)**

This enumeration specifies the values used to represent a Port's status. OK signifies the port is working properly. MARGINAL signifies errors have been experienced during recent use of the port. FAILED indicates the port is not working at all.

#### **5.3.1.1.22 PortStatusChangedEventInfo (Class)**

This class contains data that is pushed on a Field Communications event channel with a PortStatusChanged event.

#### **5.3.1.1.23 PortStatusInfo (Class)**

This class contains the data of status of a particular port.

#### **5.3.1.1.24 PortType (Class)**

This enumeration defines the types of ports that may be requested from a PortManager.

#### **5.3.1.1.25 Priority (Class)**

This enumeration specifies the priority levels used when requesting a port from a PortManager. ON\_DEMAND is given higher priority than POLLING.

#### **5.3.1.1.26 ProximityDirectionType (Class)**

This enumeration defines direction types used when comparing locations for proximity. Values for specifying same direction, opposite direction, and unknown direction are supported.

#### **5.3.1.1.27 ProximityDistance (Class)**

This struct defines members that represent proximity and distance information used when comparing locations based on position.

#### **5.3.1.1.28 ProximityInfo (Class)**

This struct defines members that represent proximity information used when comparing locations based on position.

#### **5.3.1.1.29 Service (Class)**

This interface is implemented by all services in the system that allow themselves to be shutdown externally. All implementing classes provide a means to be cleanly shutdown and can be pinged to detect if they are alive.



#### **5.3.1.1.30 Source (Class)**

This structure contains information about the source of the data being added to the system.

#### **5.3.1.1.31 SourceTypeValues (Class)**

This enumeration contains the possible sources of information that can be used for adding CommLog entries and/or traffic event data.

#### **5.3.1.1.32 SpecifiedObjectNotFound (Class)**

Exception used to indicate that an operation was attempted that involves a secondary object that cannot be found by the invoked object.

#### **5.3.1.1.33 TimeSpecificationType (Class)**

This enumeration lists the types of times which can be stored in the AbsoluteOrRelativeTime union.

#### **5.3.1.1.34 TimeStamp (Class)**

This typedef defines the type of TimeStamp fields.

#### **5.3.1.1.35 TimeStamp2 (Class)**

This data type offers extended date range beyond the year 2038 limitation implicit in the TimeStamp data type.

#### **5.3.1.1.36 TrafficParameters (Class)**

This struct contains traffic parameters that are sensed and reported by a Traffic Sensor System such as the RTMS.

m\_speedData - The arithmetic mean of the speeds collected over a sample period in miles per hour in tenths. (thus 550 == 55.0 MPH) Valid values are 0 to 2550. A value of 65535 is used to indicate a missing or invalid value (such as when the volume for the sample period is zero).

m\_volumeData - The count of vehicles for the sample period. Valid values 0 to 65535. A value of 65535 represents a missing value.

m\_percentOccupancy - The percentage of occupancy of the roadway in tenths of a percent. (thus 1000 = 100.0 percent). Valid values are 0 to 1000. A value of 65535 represents a missing or invalid value.

#### **5.3.1.1.37 UniquelyIdentifiable (Class)**

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

#### **5.3.1.1.38 UnsupportedOperation (Class)**

This exception is used to indicate that an operation is not supported by the object on which it is called.

#### **5.3.1.1.39 UserName (Class)**

This typedef defines the type of UserName fields used in system interfaces.

### 5.3.1.2 DecisionSupport (Class Diagram)

This diagram shows interfaces and structures related to decision support.

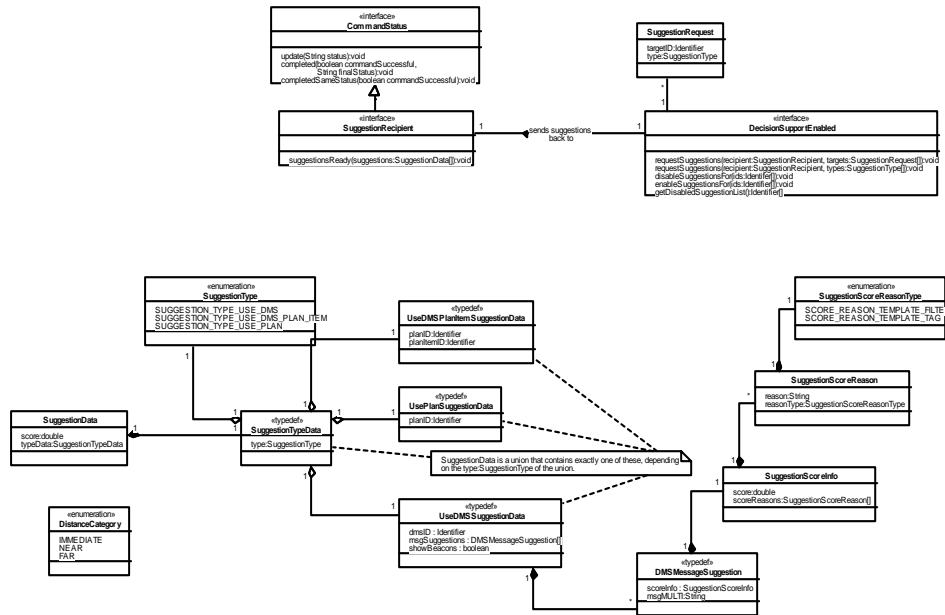


Figure 5-41. DecisionSupport (Class Diagram)

#### 5.3.1.2.1 CommandStatus (Class)

The **CommandStatus** CORBA interface is used to allow a calling process to be notified of the progress of a long-running asynchronous operation. This is normally used when field communications are involved to complete a method call. The most common use is to allow a GUI to show the user the progress of an operation. It can also be used and watched by a server process when it needs to call on another server process to complete an operation. The long running operation typically calls back to the **CommandStatus** object periodically as the command is being executed, to provide in-progress status information, and it always makes a final call to the **CommandStatus** when the operation has completed. The final call to the **CommandStatus** from the long running operation indicates the success or failure of the command.

#### 5.3.1.2.2 DecisionSupportEnabled (Class)

This class is a CORBA interface that facilitates requesting suggestions from any server side component.

#### 5.3.1.2.3 DistanceCategory (Class)

This enum represents the different Distance Types used in decision support.

#### **5.3.1.2.4 DMSMessageSuggestion (Class)**

This class is a CORBA structure that represents a decision support DMS message suggestion.

#### **5.3.1.2.5 SuggestionData (Class)**

This class is a CORBA structure that represents a decision support suggestion.

#### **5.3.1.2.6 SuggestionRecipient (Class)**

This class is a CORBA interface that must be implemented by any component that would like to request suggestions from a DecisionSupportEnabled component. This class can be used to provide suggestions back to the requestor either when they are fully formulated or by streaming them as they become available.

#### **5.3.1.2.7 SuggestionRequest (Class)**

This class is a CORBA structure that represents a request for decision support suggestions.

#### **5.3.1.2.8 SuggestionScoreInfo (Class)**

This class is a CORBA structure that represents a collection of suggestion scores/reasons for a given decision support DMS suggestion.

#### **5.3.1.2.9 SuggestionScoreReason (Class)**

This class is a CORBA structure that represents a decision support DMS message suggestion score/reason. Both scores and reasons are contained in this object.

#### **5.3.1.2.10 SuggestionScoreReasonType (Class)**

This class is an enumeration describing the 2 types of suggestion scores/reasons: template filters and template tags.

#### **5.3.1.2.11 SuggestionType (Class)**

This enumeration lists all possible types of decision support suggestions.

#### **5.3.1.2.12 SuggestionTypeData (Class)**

This class is a CORBA union that will hold a discriminator that indicates the type of data carried and exactly one of the possible SuggestionData structures.

#### **5.3.1.2.13 UseDMSPlanItemSuggestionData (Class)**

This class is a CORBA structure that represents a decision support DMS plan item suggestion.

#### **5.3.1.2.14 UseDMSSuggestionData (Class)**

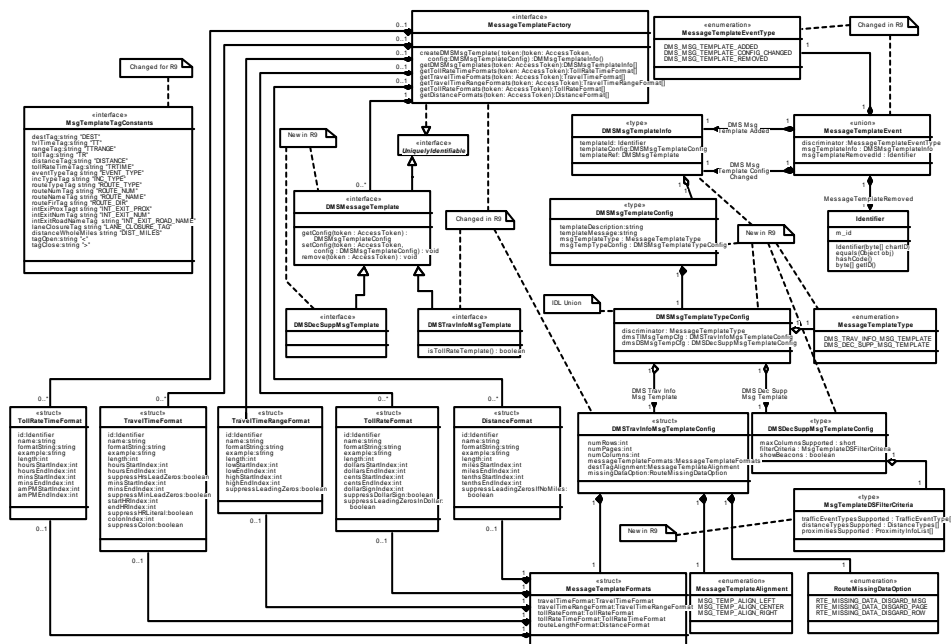
This class is a CORBA structure that represents a decision support DMS suggestion.

#### **5.3.1.2.15 UsePlanSuggestionData (Class)**

This class is a CORBA structure that represents a decision support plan suggestion.

### 5.3.1.3 MessageTemplateManagement (Class Diagram)

This class diagram contains classes that define the corba interface used for managing Message Templates in CHART2.



### Figure 5-42. MessageTemplateManagement (Class Diagram)

#### 5.3.1.3.1 DistanceFormat (Class)

This object contains the data for a distance format in the CHART DB.

#### 5.3.1.3.2 DMSDecSuppMsgTemplate (Class)

The DMSDecSuppMsgTemplate interface is implemented by objects that DMS message templates used for decision support functionality (dms message suggestion generation).

#### 5.3.1.3.3 DMSDecSuppMsgTemplateConfig (Class)

This object contains the configuration data for a message template that represents a DMSDecSuppMsgTemplate in the CHART DB

#### 5.3.1.3.4 DMSMessageTemplate (Class)

This interface is extended by interfaces representing specific types of DMS message templates (ex. DMSTravInfoMsgTemplater). It contains methods common to all types of DMS message templates.

#### **5.3.1.3.5 DMSMsgTemplateConfig (Class)**

This struct represents configuration data that is common to all DMS message template types. It contains a union that is used specify DMS message type specific configuration data.

#### **5.3.1.3.6 DMSMsgTemplateInfo (Class)**

This struct contains a DMS message configuration and a CORBA reference to the DMS message template object along with the templates unique ID.

#### **5.3.1.3.7 DMSMsgTemplateTypeConfig (Class)**

This union is used to represent DMS message template type specific configuration data in the DMSMsgTemplateConfig IDL struct.

#### **5.3.1.3.8 DMSTravInfoMsgTemplate (Class)**

The DMSTravInfoMsgTemplate interface is implemented by objects that allow execution of tasks associated with DMS travel information message templates.

#### **5.3.1.3.9 DMSTravInfoMsgTemplateConfig (Class)**

This object contains the configuration data for a message template that represents a DMSTravInfoMsgTemplate in the CHART DB

#### **5.3.1.3.10 Identifier (Class)**

Wrapper class for a CHART2 identifier byte sequence. This class will be used to add identifiable objects to hash tables and perform subsequent lookup operations.

#### **5.3.1.3.11 MessageTemplateAlignment (Class)**

This IDL enumeration defines the message template alignment options supported in the DMSTravInfoMsgTemplateConfig. These can either be align left, align center or align right.

#### **5.3.1.3.12 MessageTemplateEvent (Class)**

This union identifies the data to be passed with events that are pushed through the CORBA event service in relation to message templates.

#### **5.3.1.3.13 MessageTemplateEventType (Class)**

This IDL enumeration defines the types of CORBA Events supported in the MessageTemplateModule. These can either be DMS message Created, Changed, or Removed.



#### **5.3.1.3.14 MessageTemplateFactory (Class)**

Interface whose implementation is used to create message templates, retrieve message templates and retrieve travel time and toll rate formats.

#### **5.3.1.3.15 MessageTemplateFormats (Class)**

This structure contains all travel time and toll rate format that are specified within a given DMSTravInfoMsgTemplate.

#### **5.3.1.3.16 MessageTemplateType (Class)**

This enum represents the currently supported message template types.

#### **5.3.1.3.17 MsgTemplateDSFilterCriteria (Class)**

This IDL struct contains criteria used for filtering when finding applicable devices to use with a message template in decision support.

#### **5.3.1.3.18 MsgTemplateTagConstants (Class)**

The MessageTemplateTagConstants interface contains constants that are used to define tags used to create DMS travel information message templates.

#### **5.3.1.3.19 RouteMissingDataOption (Class)**

This IDL enumeration defines the route missing data options supported in the DMSTravInfoMsgTemplateConfig. These can either be discard row, discard page or discard row.

#### **5.3.1.3.20 TollRateFormat (Class)**

This object contains the data for a toll rate format in the CHART DB.

#### **5.3.1.3.21 TollRateTimeFormat (Class)**

This object contains the data for a toll rate time format in the CHART DB.

#### **5.3.1.3.22 TravelTimeFormat (Class)**

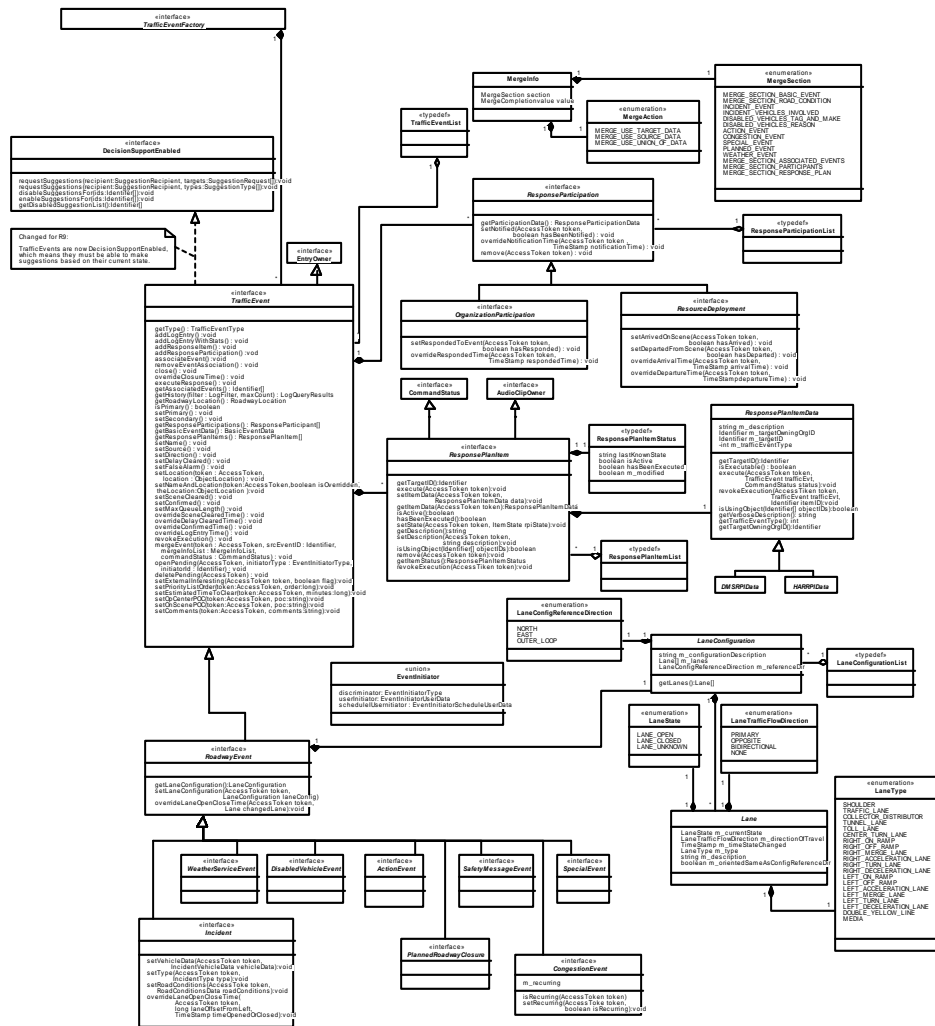
This object contains the data for a travel time format in the CHART DB.

#### **5.3.1.3.23 TravelTimeRangeFormat (Class)**

This object contains the data for a travel time range format in the CHART DB.

#### 5.3.1.4 TrafficEventManager (Class Diagram)

This class diagram contains all classes relating to Traffic Events



**Figure 5-43. TrafficEventManager (Class Diagram)**

#### 5.3.1.4.1 ActionEvent (Class)

This class models roadway events that require an operations center to take action but do not fit well into the other event categories. An example of this type of event would be debris in the roadway.

#### **5.3.1.4.2 AudioClipOwner (Class)**

This interface allows the AudioClipManager to check whether there are any parties interested in an audio clip. If no AudioClipOwners claim interest in a clip, the clip can be deleted.

#### **5.3.1.4.3 CommandStatus (Class)**

The CommandStatus CORBA interface is used to allow a calling process to be notified of the progress of a long-running asynchronous operation. This is normally used when field communications are involved to complete a method call. The most common use is to allow a GUI to show the user the progress of an operation. It can also be used and watched by a server process when it needs to call on another server process to complete an operation. The long running operation typically calls back to the CommandStatus object periodically as the command is being executed, to provide in-progress status information, and it always makes a final call to the CommandStatus when the operation has completed. The final call to the CommandStatus from the long running operation indicates the success or failure of the command.

#### **5.3.1.4.4 CongestionEvent (Class)**

This class models roadway congestion which may be tagged as recurring or non-recurring through the use of an attribute.

#### **5.3.1.4.5 DecisionSupportEnabled (Class)**

This class is a CORBA interface that facilitates requesting suggestions from any server side component.

#### **5.3.1.4.6 DisabledVehicleEvent (Class)**

This class models disabled vehicles on the roadway.

#### **5.3.1.4.7 DMSRPIData (Class)**

The DMSRPIData class is an abstract class which describes a response plan item for a DMS. It contains the unique identifier of the DMS to contain the DMSMessage, and the DMSMessage itself.

#### **5.3.1.4.8 EntryOwner (Class)**

Interface which must be implemented by any class which is responsible for putting an ArbQueueEntry on a device's arbitration queue. This validate method of this interface can be called by the device to determine continued validity of the entry (either during recovery or as a final check of the validity of an entry before putting its message on the device).

#### **5.3.1.4.9 EventInitiator (Class)**

This union contains information about the entity or entities involved in the initiation of a traffic event. This can be the schedule, if a schedule was involved in initiating the event, and/or a user, if a user was involved in initiating the event. This union allows for possible expansion in future releases, where traffic events may be initiated by a schedule without user confirmation, or by CHART devices (traffic sensors, weather sensors, etc.) or external interfaces (RITIS, etc.) initially with, or possibly later without, user involvement.

#### **5.3.1.4.10 HARRPIData (Class)**

This class represents an item in a traffic event response plan that is capable of issuing a command to put a message on a HAR when executed. When the item is executed, it adds an ArbQueueEntry to the specified HAR, which stores the entry in its MessageQueue. When the item's execution is revoked, or the item is removed from the response plan (manually or implicitly through closing the traffic event) the item asks the HAR to remove the entry. The HARRPIData object also allows specification of a subset (0 to all) of the HARNotifier devices (SHAZAM or DMS devices acting as SHAZAMs) to be activated if and while the message is being broadcast on the HAR.

#### **5.3.1.4.11 Incident (Class)**

This class models objects representing roadway incidents. An incident typically involves one or more vehicles and roadway lane closures.

#### **5.3.1.4.12 Lane (Class)**

This class represents a single traffic lane at the scene of a RoadwayEvent.

#### **5.3.1.4.13 LaneConfigReferenceDirection (Class)**

This enumeration restricts the possible reference directions for a lane configuration, which is necessary because the lane offsets are defined relative to the "left" side, which is an ambiguous term. For example, if the direction is North then "left" to the West, but if the direction is South (also valid on a North-South roadway) then "left" could be considered (if not for this enumeration) to East. Thus if the direction of the lane config were to change from North to South, the lanes would "flip" unintentionally. This enumeration holds the reference direction for a North-South roadway to always be to the West (regardless of whether the direction of the event is North or South), and holds similarly for East-West roadways and beltways (Inner-Outer loops).

#### **5.3.1.4.14 LaneConfiguration (Class)**

This class contains data that represents the configuration of the lanes.

#### **5.3.1.4.15 LaneConfigurationList (Class)**

A collection of LaneConfiguration objects.

#### **5.3.1.4.16 LaneState (Class)**

This enumeration lists the possible states that a traffic lane may be in.

#### **5.3.1.4.17 LaneTrafficFlowDirection (Class)**

Defines the possible directions of traffic flow, relative to the lane orientation.

#### **5.3.1.4.18 LaneType (Class)**

This enumeration lists the types of lanes.

#### **5.3.1.4.19 MergeAction (Class)**

This enumeration specifies how to merge a section of data during a traffic event merge operation.

#### **5.3.1.4.20 MergeInfo (Class)**

This valuetype is passed between Chartlite to Chart to provide instructions for performing the merge

#### **5.3.1.4.21 MergeSection (Class)**

This idl enum defines values for each merge section

#### **5.3.1.4.22 OrganizationParticipation (Class)**

This class is used to manage the data captured when an operator notifies another organization of a traffic event.

#### **5.3.1.4.23 PlannedRoadwayClosure (Class)**

This class models planned roadway closures such as road construction. This interface will be expanded in future releases to include interfacing with the EORS system.

#### **5.3.1.4.24 ResourceDeployment (Class)**

This class is used to store the data captured when an operator deploys resources to the scene of a traffic event.

#### **5.3.1.4.25 ResponseParticipation (Class)**

This interface represents the involvement of one particular resource or organization in response to a particular traffic event.

#### **5.3.1.4.26 ResponseParticipationList (Class)**

A collection of ResponseParticipation objects.

#### **5.3.1.4.27 ResponsePlanItem (Class)**

Objects of this type can be executed as part of a traffic event response plan. A ResponsePlanItem can be executed by an operator, at which time it becomes the responsibility of the System to activate the item on the ResponseDevice as soon as it is appropriate.

#### **5.3.1.4.28 ResponsePlanItemData (Class)**

This class is a delegate used to perform the execute and remove tasks for the response plan item. Derived classes of this base class have specific implementations for the type of device the response plan item is used to control.

#### **5.3.1.4.29 ResponsePlanItemList (Class)**

A collection of ResponsePlanItem objects.

#### **5.3.1.4.30 ResponsePlanItemStatus (Class)**

This structure contains data that describes the current state of a response plan item.

#### **5.3.1.4.31 RoadwayEvent (Class)**

This class models any type of incident that can occur on a roadway. This point in the hierarchy provides a break off point for traffic event types that pertain to other modals.

#### **5.3.1.4.32 SafetyMessageEvent (Class)**

This type of event is created by an operator when he/she would like to send a safety message to a device.

#### **5.3.1.4.33 SpecialEvent (Class)**

This class models special events that affect roadway conditions such as a concert or professional sporting event.

#### **5.3.1.4.34 TrafficEvent (Class)**

Objects of this type represent traffic events that require action from system operators.

#### **5.3.1.4.35 TrafficEventFactory (Class)**

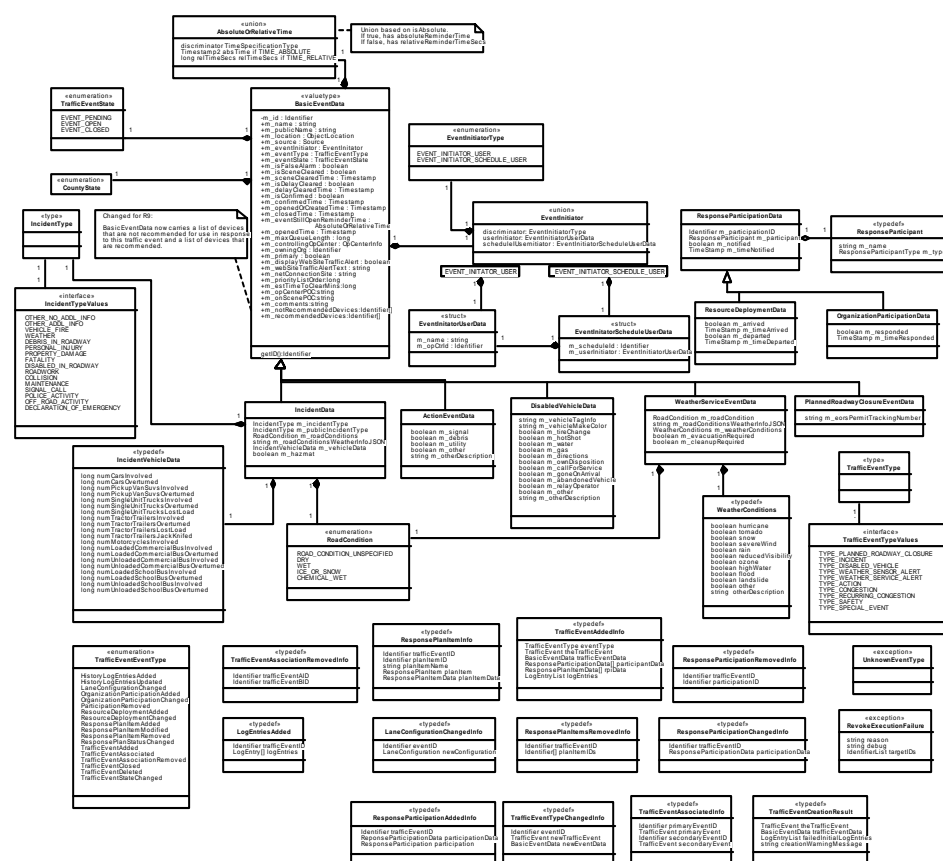
This interface is supported by objects that are capable of creating traffic event objects in the system.

#### **5.3.1.4.36 WeatherServiceEvent (Class)**

This class models roadway weather events such as snow or fog that are manually entered by an operator in response to receiving an alert from the national weather service.

### 5.3.1.5 TrafficEventManager2 (Class Diagram)

This diagram shows more interface classes related to traffic events.



**Figure 5-44. TrafficEventManager2 (Class Diagram)**

#### 5.3.1.5.1 AbsoluteOrRelativeTime (Class)

This union stores a time, in either absolute or relative terms.

#### 5.3.1.5.2 ActionEventData (Class)

This class represents all data specific to an Action event type traffic event.

#### 5.3.1.5.3 BasicEventData (Class)

This class represents the data common to all traffic events. All derived data types will inherit all data shown in this class.

#### **5.3.1.5.4 CountyState (Class)**

This enumeration defines the various counties in Maryland and the states surrounding Maryland that will be used for defining the traffic event.

#### **5.3.1.5.5 DisabledVehicleData (Class)**

This class represents all data specific to a disabled vehicle traffic event.

#### **5.3.1.5.6 EventInitiatorScheduleUserData (Class)**

This structure contains data about a schedule involved in the initiation of a traffic event. It is contained within the EventInitiator union.

#### **5.3.1.5.7 EventInitiatorUserData (Class)**

This structure contains data about a user involved in the initiation of a traffic event. It is contained within the EventInitiator union.

#### **5.3.1.5.8 EventInitiator (Class)**

This union contains information about the entity or entities involved in the initiation of a traffic event. This can be the schedule, if a schedule was involved in initiating the event, and/or a user, if a user was involved in initiating the event. This union allows for possible expansion in future releases, where traffic events may be initiated by a schedule without user confirmation, or by CHART devices (traffic sensors, weather sensors, etc.) or external interfaces (RITIS, etc.) initially with, or possibly later without, user involvement.

#### **5.3.1.5.9 EventInitiatorType (Class)**

This enumeration identifies the types of initiators which can initiate traffic events. Traffic events can be initiated by a user (directly), or by a schedule (with user involvement). This enumeration, and the union in which it is a discriminator, allows for possible expansion in future releases, where traffic events may be initiated by a schedule without user confirmation, or by CHART devices (traffic sensors, weather sensors, etc.) or external interfaces (RITIS, etc.) initially with, or possibly later without, user involvement.

#### **5.3.1.5.10 IncidentData (Class)**

This class represents data specific to an Incident type traffic event.

#### **5.3.1.5.11 IncidentType (Class)**

This typedef defines the type of the incident.

#### **5.3.1.5.12 IncidentTypeValues (Class)**

This interface lists all possible incident types.



#### **5.3.1.5.13 IncidentVehicleData (Class)**

This class represents the vehicles involved data for incidents. Its purpose is to simplify the exchange of data between GUI and server.

#### **5.3.1.5.14 LaneConfigurationChangedInfo (Class)**

This structure contains the data that is broadcast when the lane configuration of a traffic event is changed.

#### **5.3.1.5.15 LogEntriesAdded (Class)**

This structure contains the data that is broadcast when new entries are added to the event history log of a traffic event.

#### **5.3.1.5.16 OrganizationParticipationData (Class)**

This class represents the data required to describe an organization's participation in the response to a traffic event.

#### **5.3.1.5.17 PlannedRoadwayClosureEventData (Class)**

This class contains data specific to the PlannedRoadwayEvent type of traffic event.

#### **5.3.1.5.18 ResourceDeploymentData (Class)**

This class represents the data required to describe a resource's participation in the response to a traffic event.

#### **5.3.1.5.19 ResponseParticipant (Class)**

The ResponseParticipant class is a non-behavioral structure which specifies a participant in a response.

#### **5.3.1.5.20 ResponseParticipationAddedInfo (Class)**

This structure contains the data that is broadcast when a response participant is added to the response to a particular traffic event.

#### **5.3.1.5.21 ResponseParticipationChangedInfo (Class)**

This structure contains the data pushed in a CORBA event any time any type of response participation object changes state.

#### **5.3.1.5.22 ResponseParticipationData (Class)**

This class contains all data pertinent to any class that represents a response participation.

#### **5.3.1.5.23 ResponseParticipationRemovedInfo (Class)**

This structure contains the data that is broadcast when one or more response plan items are removed from a traffic event.

#### **5.3.1.5.24 ResponsePlanItemInfo (Class)**

This structure contains the data that is broadcast any time a new response plan item is added or an existing response plan item is modified.

#### **5.3.1.5.25 ResponsePlanItemsRemovedInfo (Class)**

This structure contains the data that is broadcast when one or more response plan items are removed from a traffic event.

#### **5.3.1.5.26 RevokeExecutionFailure (Class)**

This class defines a exception thrown when failed to revoke a response plan item's execution.

#### **5.3.1.5.27 RoadCondition (Class)**

This enumeration lists the possible roadway conditions at the scene of a traffic event.

#### **5.3.1.5.28 TrafficEventAddedInfo (Class)**

This structure contains the data that is broadcast when a new traffic event is added to the system.

#### **5.3.1.5.29 TrafficEventAssociatedInfo (Class)**

This structure contains the data that is broadcast when two traffic events are associated.

#### **5.3.1.5.30 TrafficEventAssociationRemovedInfo (Class)**

This structure contains the data that is broadcast when the association between two traffic events is removed.

#### **5.3.1.5.31 TrafficEventCreationResult (Class)**

This result is returned from createEvent() to indicate warning messages if the event was not created cleanly.

#### **5.3.1.5.32 TrafficEventEventType (Class)**

This enumeration defines the types of CORBA events that can be broadcast on a Traffic Event related CORBA Event channel.

#### **5.3.1.5.33 TrafficEventState (Class)**

This enumeration lists the possible states for a traffic event. The states are pending, open, and closed. A false alarmed "state" is considered a special case of "closed", so false alarmed events will have a TrafficEventState of EVENT\_STATE\_CLOSED. They will also have the m\_isFalseAlarm flag in their BasicEventData set to true to distinguish them from normally closed events.

#### **5.3.1.5.34 TrafficEventType (Class)**

This typedef defines the type of traffic event.

#### **5.3.1.5.35 TrafficEventTypeChangedInfo (Class)**

This structure contains the data that is broadcast when a traffic event changes types. The traffic event object that represented the traffic event previously is removed from the system and is replaced by the newTrafficEvent reference contained in this structure. If the consumer of this CORBA event has stored any references to the traffic event previously, those references should be replaced with this new reference.

#### **5.3.1.5.36 TrafficEventTypeValues (Class)**

This interface defines the types of traffic events that are supported by the system.

#### **5.3.1.5.37 UnknownEventType (Class)**

This class defines a exception thrown when the type of a traffic event type is not known and is not defined in TrafficEventTypeValues.

#### **5.3.1.5.38 WeatherConditions (Class)**

This structure contains all possible weather conditions. Each member should be set to true if that condition applies, false otherwise. The m\_otherDescription member will only be considered valid if the m\_other member is set to true.

#### **5.3.1.5.39 WeatherServiceEventData (Class)**

This class contains data specific to the WeatherServiceEvent type of traffic event.

## 5.4 TrafficEventModule

### 5.4.1 Class Diagrams

#### 5.4.1.1 TrafficEventModuleClasses2 (Class Diagram)

This diagram shows Traffic Event Module classes related to the decision support changes in Release 9.

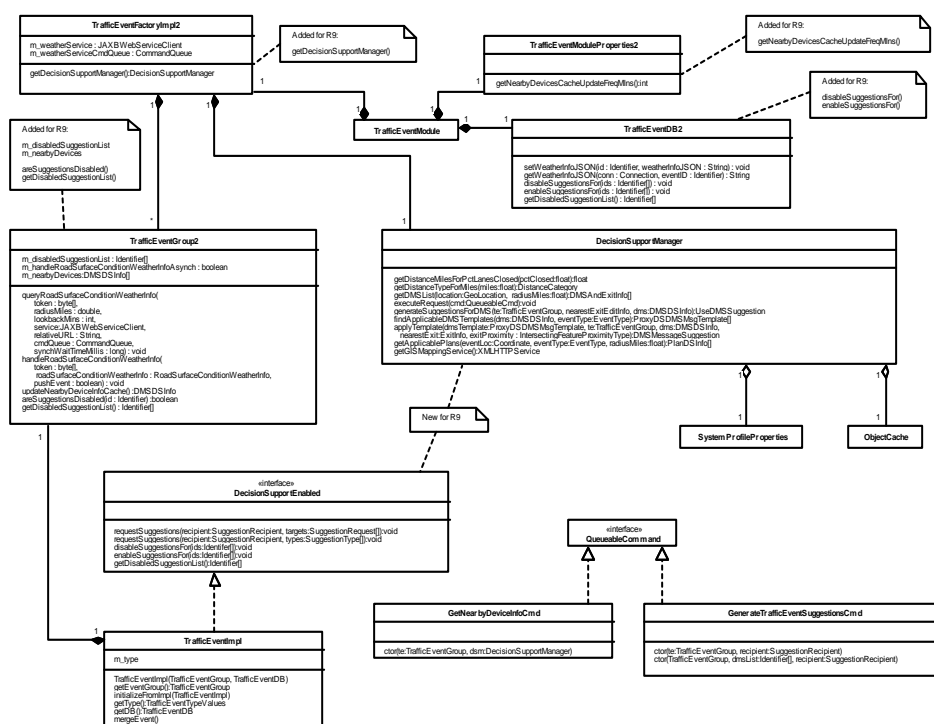


Figure 5-45. TrafficEventModuleClasses2 (Class Diagram)

##### 5.4.1.1.1 DecisionSupportEnabled (Class)

This class is a CORBA interface that facilitates requesting suggestions from any server side component.

##### 5.4.1.1.2 DecisionSupportManager (Class)

This class provides utility methods that are applicable to decision support. It includes methods for generating suggestions, applying templates, determining distances, etc.

##### 5.4.1.1.3 GenerateTrafficEventSuggestionsCmd (Class)

This class performs the asynchronous task of generating suggestions for a traffic event.

#### **5.4.1.1.4 GetNearbyDeviceInfoCmd (Class)**

This class performs the asynchronous task of determining which devices are near (within a configurable distance of) a traffic event.

#### **5.4.1.1.5 ObjectCache (Class)**

The ObjectCache is a wrapper for the DataModel. It provides access to DataModel methods to find objects in the data model, delegating those methods to the DataModel itself. It also provides additional methods of finding name filtered objects and discovering "duplicate" objects (as defined by an isDuplicateOf() method of the Duplicatable interface).

#### **5.4.1.1.6 QueueableCommand (Class)**

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

#### **5.4.1.1.7 SystemProfileProperties (Class)**

This class is used to cache the system profile properties and provide access to them. It is also used to interact with the server to change system profile settings.

#### **5.4.1.1.8 TrafficEventDB2 (Class)**

This class contains TrafficEventDB changes for R7. This class provides database functionality related to traffic events.

#### **5.4.1.1.9 TrafficEventFactoryImpl2 (Class)**

This class shows changes in the TrafficEventFactoryImpl class for R7. This class manages all traffic events served by the Traffic Event Service.

#### **5.4.1.1.10 TrafficEventGroup2 (Class)**

This class shows changes in the TrafficEventGroup class for R7. This class manages a single TrafficEvent CORBA object implementation.

#### **5.4.1.1.11 TrafficEventImpl (Class)**

This class provides an implementation of the TrafficEvent interface. It contains state variables and processing that common to all traffic events.

#### **5.4.1.1.12 TrafficEventModule (Class)**

This class provides the resources and support functionality necessary to serve traffic event related objects in a service application. It implements the ServiceApplicationModule interface which allows it to be served from any ServiceApplication.

#### **5.4.1.1.13 TrafficEventModuleProperties2 (Class)**

This class contains TrafficEventModuleProperties changes for R7. This class provides access to settings defined in the TrafficEventService properties file.

## 5.4.2 Sequence Diagrams

### 5.4.2.1 TrafficEventGroup:disableSuggestionsFor (Sequence Diagram)

This diagram shows the processing that occurs when the TrafficEventImpl object is called to disable suggestions for a list of devices (either DMSs or Plans). The login session access token is checked to verify the user has the functional rights to manage the traffic event. The ID of the traffic event and the list of IDs for the devices is passed to the traffic event database object and used to create delete and insert statements for each of the devices. The database changes are processed in a transaction in case any of the changes fail. Once the database has been successfully updated, the ID for each of the devices is added to the list of disabled devices in the traffic event group.

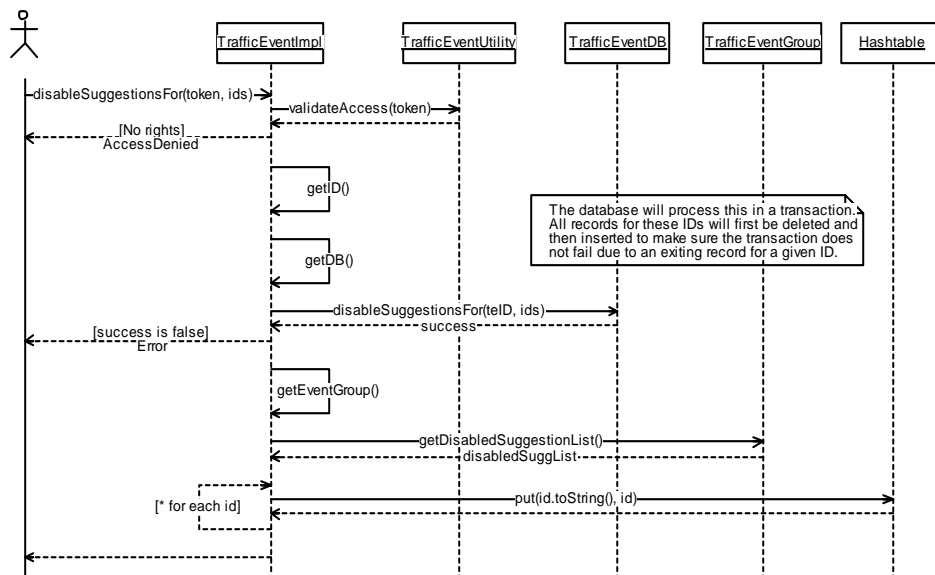
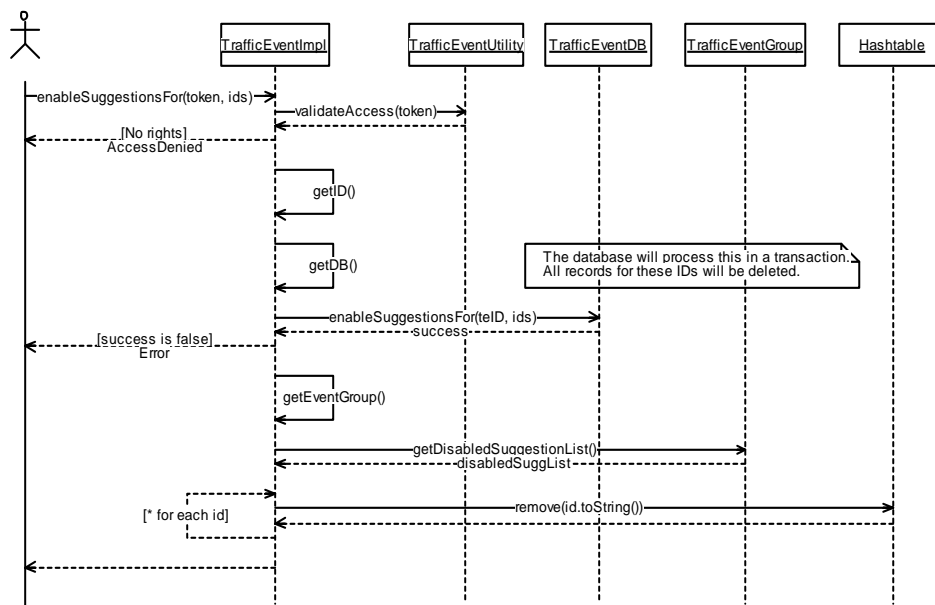


Figure 5-46. TrafficEventGroup:disableSuggestionsFor (Sequence Diagram)

### 5.4.2.2 TrafficEventGroup:enableSuggestionsFor (Sequence Diagram)

This diagram shows the processing that occurs when the TrafficEventImpl object is called to enable suggestions for a list of devices (either DMSs or Plans). The login session access token is checked to verify the user has the functional rights to manage the traffic event. The ID of the traffic event and the list of IDs for the devices is passed to the traffic event database object and used to create delete statements for each of the devices. The database changes are processed in a transaction in case any of the changes fail. Once the database has been successfully updated, the ID for each of the devices is removed from the list of disabled devices in the traffic event group.



**Figure 5-47. TrafficEventGroup:enableSuggestionsFor (Sequence Diagram)**



### 5.4.2.3 TrafficEventGroup:requestSuggestions (Sequence Diagram)

This diagram shows the processing that occurs when a SuggestionRecipient object is to request suggestions for a DMS. The login session is checked to verify the user has the functional rights to manage the response plan. The traffic event is checked to verify it is not closed. A command object (GenerateTrafficEventSuggestionsCmd) is created and asynchronously executed by the Command Queue. The command object requests suggestions from the Traffic Event Group and passes the Suggestion Recipient to receive the results. The Traffic Event Group determines the direction of the event, the percentage of lanes closed, the distance type, and the location of the event. It then gets a list of DMSs within a configurable radius. For each DMS in the list, the Decision Support Manager is used to create suggestions including: DMS suggestions, DMS plan item suggestions, and plan suggestions. All of the resulting suggestions are passed to the SuggestionRecipient object.

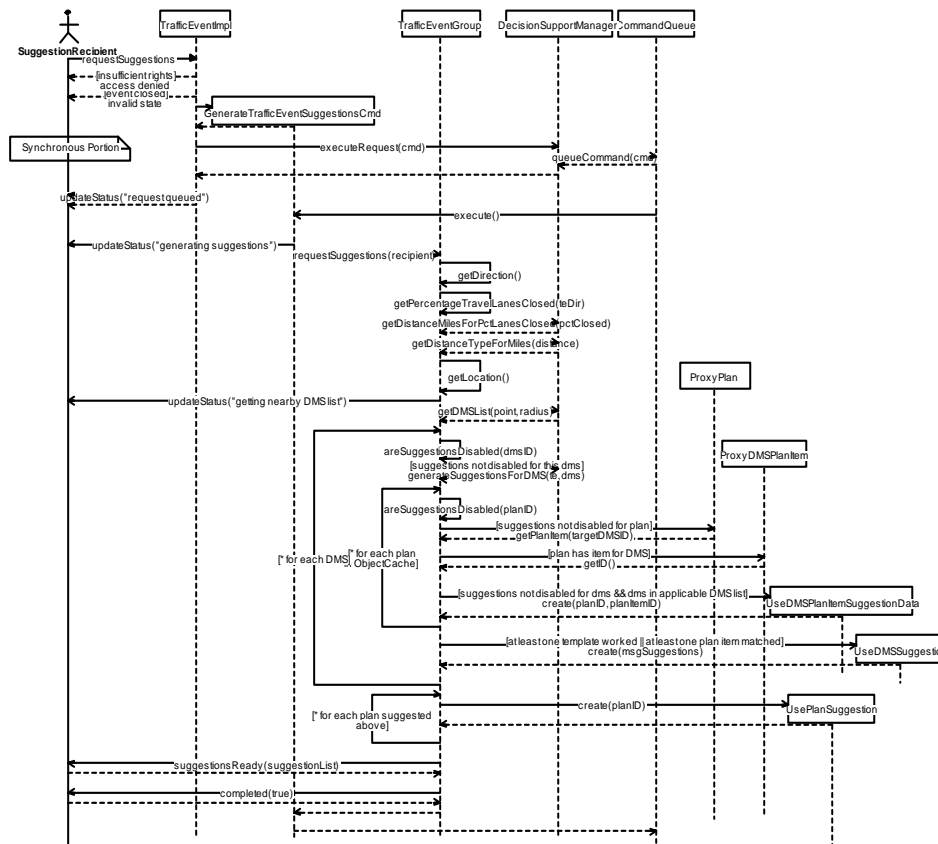


Figure 5-48. TrafficEventGroup:requestSuggestions (Sequence Diagram)

#### 5.4.2.4 TrafficEventModule2:initialize (Sequence Diagram)

This diagram shows the processing that occurs when the TrafficEventModule object is initialized. The TrafficEventModuleProperties object is created and the frequency (in minutes) to update the nearby devices cache is determined. The TrafficEventDB object is created. The TrafficEventFactoryImpl object is created. The GISMappingService object is created. The discovery manager is used to update the system profile properties. A command to update object cache with ProxyDMS objects (DiscoverDMSClassesCmd) is created and added to the discovery manager. A command to update object cache with ProxyPlan objects (DiscoverPlanClassesCmd) is created and added to the discovery manager. A command to update object cache with ProxyDMSMsgTemplate objects (DiscoverDMSMsgTemplateClassesCmd) is created and added to the discovery manager. A Timer object is created for use in scheduling timer tasks. A command object for updating the cache with nearby devices information (GetNearbyDeviceInfoCmd) is created and scheduled based on the frequency (in minutes) to update the nearby devices cache.

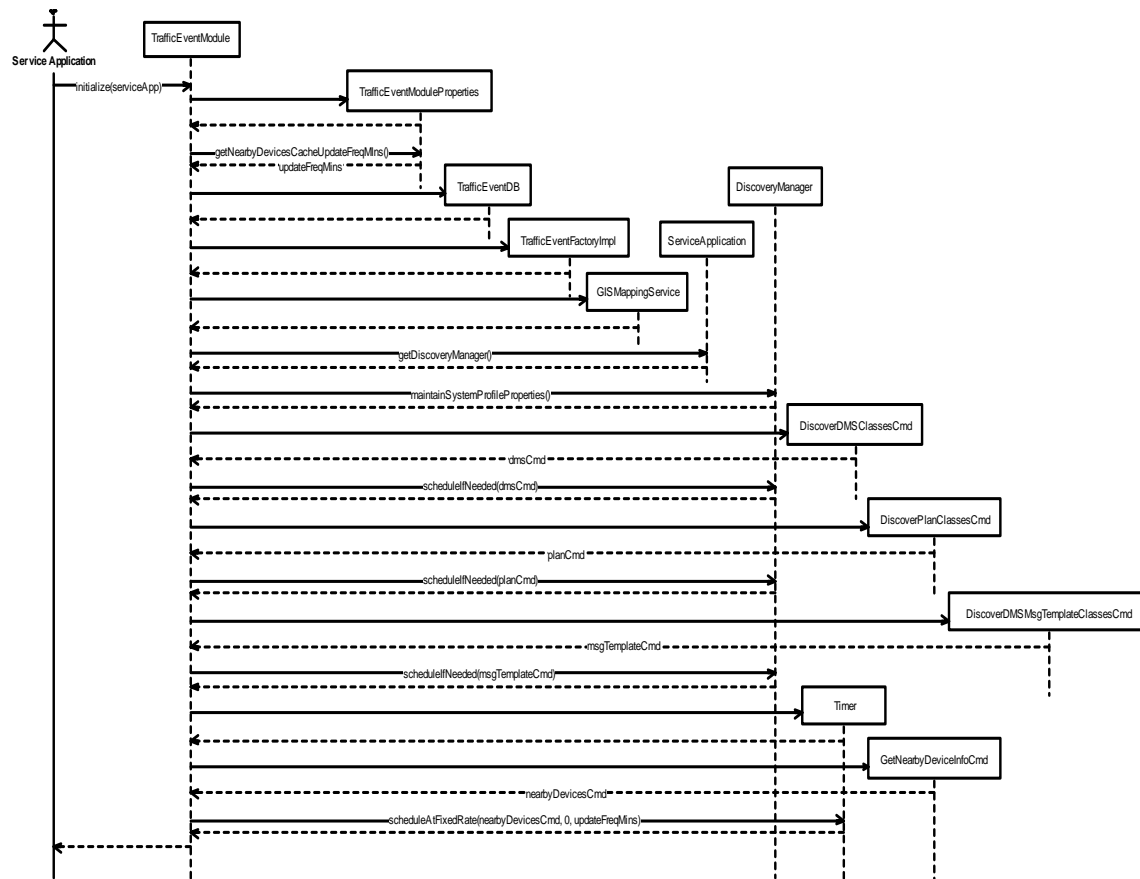


Figure 5-49. TrafficEventModule2:initialize (Sequence Diagram)

## 5.5 DecisionSupportUtility

### 5.5.1 Class Diagrams

#### 5.5.1.1 DecisionSupportUtility (Class Diagram)

This diagram shows structures related to utility methods for decision support.

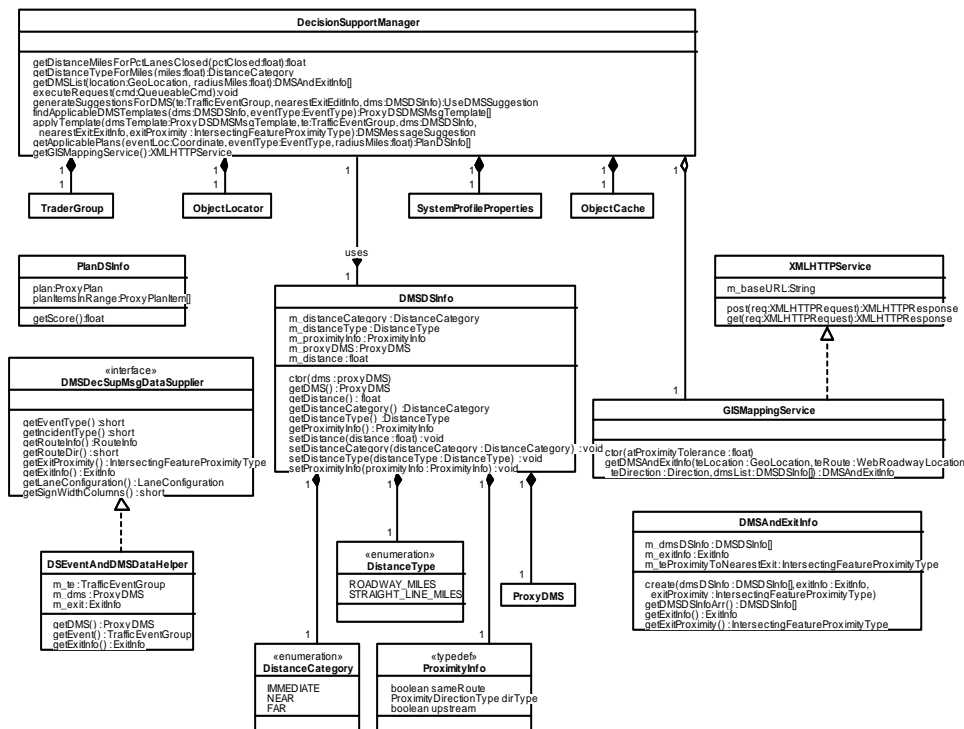


Figure 5-50. DecisionSupportUtility (Class Diagram)

##### 5.5.1.1.1 DecisionSupportManager (Class)

This class provides utility methods that are applicable to decision support. It includes methods for generating suggestions, applying templates, determining distances, etc.

##### 5.5.1.1.2 DistanceCategory (Class)

This enum represents the different Distance Types used in decision support.

##### 5.5.1.1.3 DistanceType (Class)

This enumeration represents different values for types of distance stated in miles (roadway miles, straight line miles).

#### **5.5.1.1.4 DMSAndExitInfo (Class)**

This class contains proximity information for proxy DMSs including the roadway distance (in miles) between the DMS location and the traffic event location. This class also contains information about the nearest exit to the traffic event that is located on the same route and in the same direction.

#### **5.5.1.1.5 DMSDecSupMsgDataSupplier (Class)**

This interface is implemented by objects that will supply decision support data for the purpose of generating a DMS message template suggestion. The data supplied could be "dummy" data which would be the case for editing a decision support dms message template. The data represents a DMS / Event pair.

#### **5.5.1.1.6 DMSDSInfo (Class)**

This class is a CORBA structure that represents a DMS that is suggested by decision support. This class includes distance and proximity information for the DMS.

#### **5.5.1.1.7 DSEventAndDMSDataHelper (Class)**

This class implements the DMSDecSupMsgDataSupplier interface and is a helper class used when generating DMS message suggestions. It wraps traffic event, DMS and exit data and provides methods to return that data in common ways primarily for use in the DMSDecSupMsgTemplateModel.

#### **5.5.1.1.8 GISMappingService (Class)**

This class extends the XMLHTTPService and provides specific functionality for Decision Support.

#### **5.5.1.1.9 ObjectCache (Class)**

The ObjectCache is a wrapper for the DataModel. It provides access to DataModel methods to find objects in the data model, delegating those methods to the DataModel itself. It also provides additional methods of finding name filtered objects and discovering "duplicate" objects (as defined by an isDuplicateOf() method of the Duplicatable interface).

#### **5.5.1.1.10 ObjectLocator (Class)**

This class is used to provide access to proxy objects stored in the CHART object cache (which have been discovered by the DiscoveryDriver tasks).

#### **5.5.1.1.11 PlanDSInfo (Class)**

This class is a CORBA structure that represents a plan that is suggested by decision support.

#### **5.5.1.1.12 ProximityInfo (Class)**

This struct defines members that represent proximity information used when comparing locations based on position.

#### **5.5.1.1.13 ProxyDMS (Class)**

The ProxyChart2DMS object is a proxy for a Chart2DMS corba object which is used to by the DiscoveryManager / ObjectCache. The objects are used to maintain an up to date cache of Chart2DMS data in the object cache for application use.

#### **5.5.1.1.14 SystemProfileProperties (Class)**

This class is used to cache the system profile properties and provide access to them. It is also used to interact with the server to change system profile settings.

#### **5.5.1.1.15 TraderGroup (Class)**

This class provides a facade for trader lookups that allows application level code to be unaware of the number of CORBA trading services that the application is using or the details of the linkage between those services.

#### **5.5.1.1.16 XMLHTTPService (Class)**

This class represents a remote XML/HTTP based web service at a specified URL. It supports operations to perform HTTP get and post operations on the remote service.

## 5.5.2 Sequence Diagrams

### 5.5.2.1 DecisionSupportManager:applyTemplate (Sequence Diagram)

This diagram describes the DecisionSupportManager.applyTemplate() method. The method takes data for a specific template, traffic event, dms and exit and uses it to create a model of the template which is used to generate a message suggestion.

DMSDecSupMsgTemplateModel.formatMulti() will attempt to create a multi string which can be returned as a DMSMessageSuggestion to the caller. Note: the formatMulti() method may throw a DataUnavailableException if data is not available for a tag, either because the traffic event does not support the data or the system configuration is such that a specific piece of data should not be displayed (ex. route name never to be included in suggestion content). The method returns a DMSuggestion which contains the multi string and also scoring information for the suggestion.

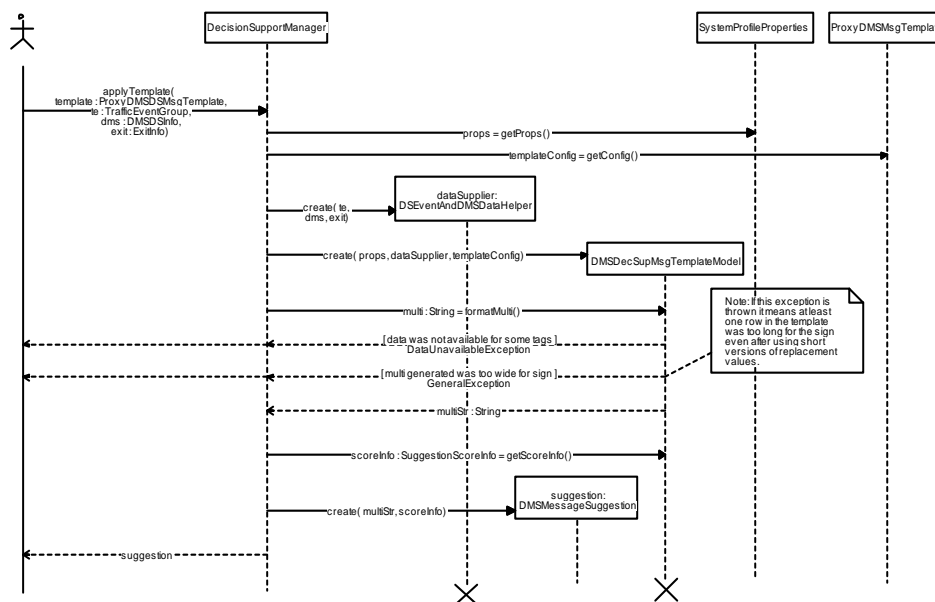


Figure 5-51. DecisionSupportManager:applyTemplate (Sequence Diagram)

### 5.5.2.2 DecisionSupportManager:generateSuggestionsForDMS (Sequence Diagram)

This diagram shows the processing that occurs when the DecisionSupportManager object is called to generate suggestions for a DMS. The TrafficEventObject is called to get the type of the event. The DMS, type of the traffic event, and the distance type (Immediate, Near, or Far) is used to find DMS templates that are applicable. Each template is applied to the DMS and traffic event. If at least one of the templates is successfully applied, a UseDMSSuggestion object is created. After all templates have been processed, all UseDMSSuggestion objects that have been created are passed to the caller.

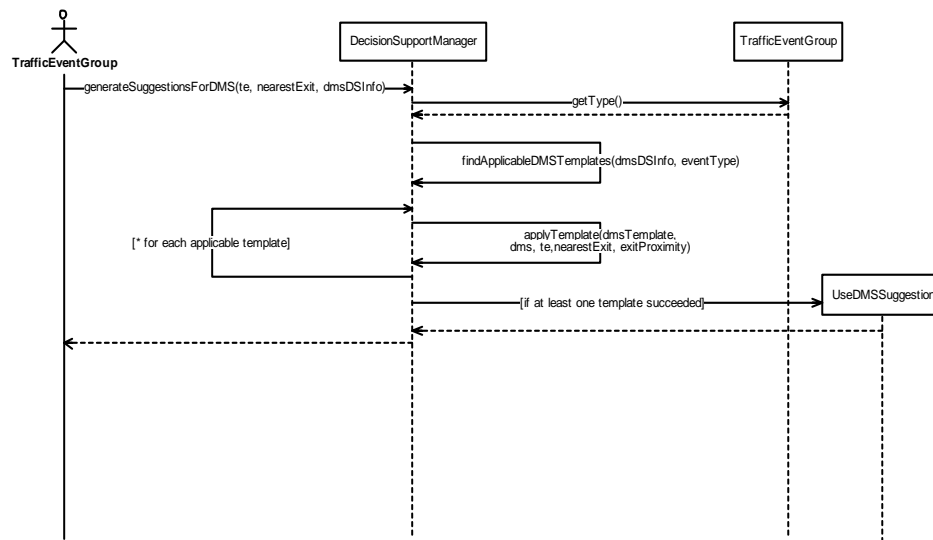


Figure 5-52. DecisionSupportManager:generateSuggestionsForDMS (Sequence Diagram)

### 5.5.2.3 DecisionSupportManager:getDMSList (Sequence Diagram)

This diagram shows the processing that occurs when the traffic event group requests information about DMSs from the decision support manager. The proxy DMSs are obtained from the object cache of the decision support manager. Two lists are created to hold proximity information about the DMSs: one for DMSs on the same route as the traffic event and one for all other DMSs. Each DMS is checked to determine if it is a CHART DMS (i.e. not an external DMS), has a geo-location, and is located within the specified radius of the traffic event. For each DMS that meets these conditions, a DMSDSInfo object is created and updated with proximity information. The DMSDSInfo object is then added to the appropriate list (i.e. it is added to the same route list if the DMS is on the same route as the traffic event; otherwise, it is added to the other list). When all DMSs have been processed, the list of DMSDSInfo objects for the same route DMSs is passed to the GIS Mapping Service to update the proximity information for each DMS with the upstream indicator. The GIS Mapping Service also provides information on the nearest exit to the traffic event. This information is returned to the decision support manager in a DMSAndExitInfo object. The updated information for same route DMSs is appended to the information for all other DMSs to create a single array of DMS information for all DMSs. This information along with the exit information is returned to the traffic event group in a DMSAndExitInfo object.

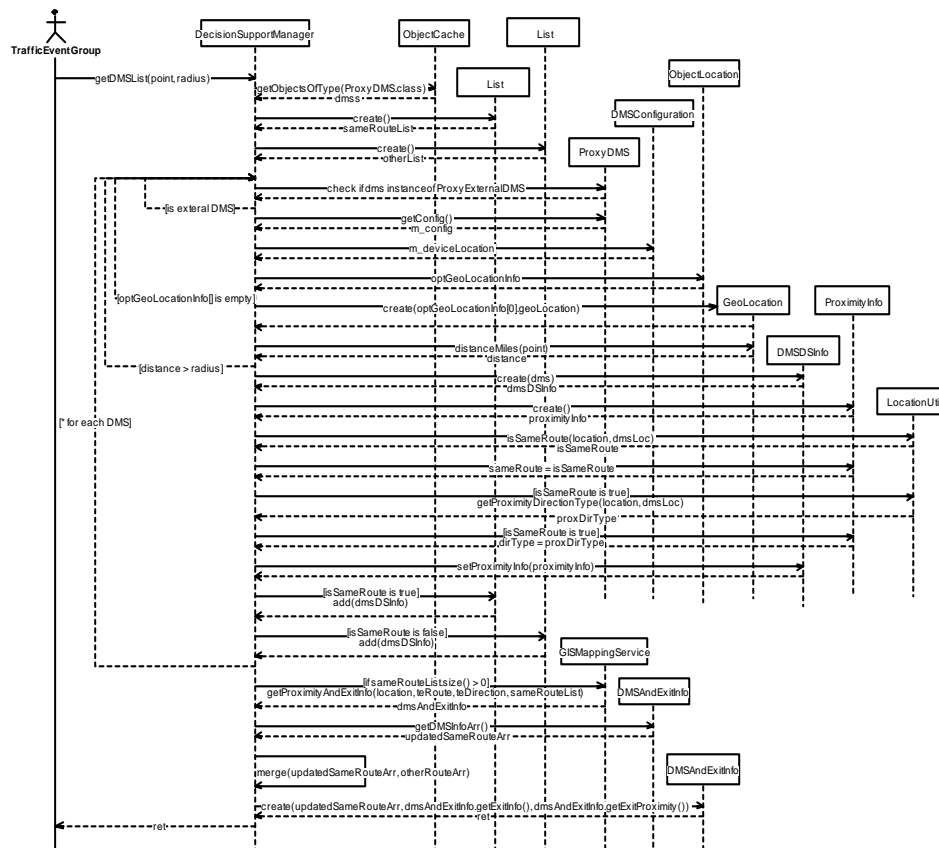


Figure 5-53. DecisionSupportManager:getDMSList (Sequence Diagram)



## 5.6 CHART2.Utility.ObjectCache

### 5.6.1 Class Diagrams

#### 5.6.1.1 ObjCachePlanRelatedClasses (Class Diagram)

This class diagram contains classes used for caching data in the object cache as proxies for Plan related objects in CHART2.

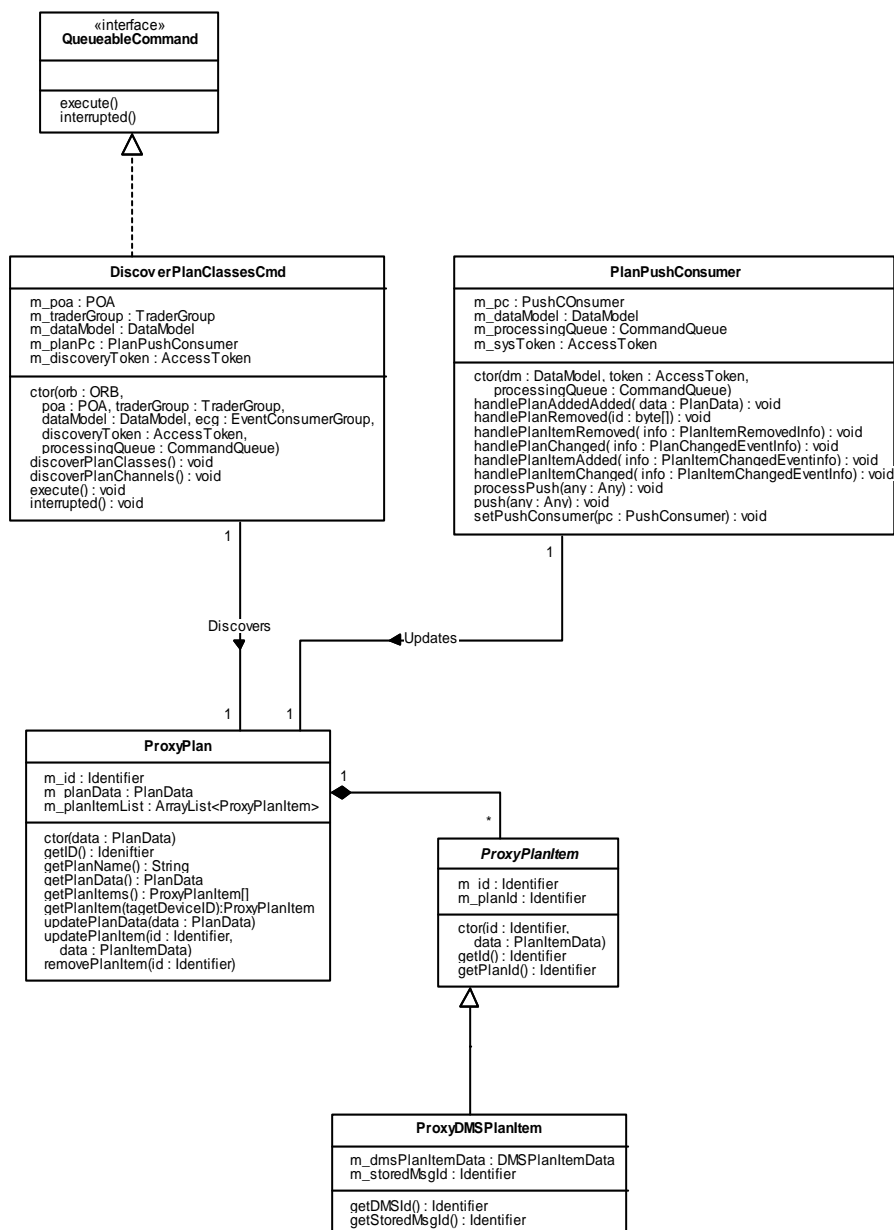


Figure 5-54. ObjCachePlanRelatedClasses (Class Diagram)

#### **5.6.1.1.1 DiscoverPlanClassesCmd (Class)**

This class is responsible for the discovery and ongoing updating of plan related proxy objects in the ObjectCache.

#### **5.6.1.1.2 PlanPushConsumer (Class)**

This class is responsible for keeping plan related proxy objects updated in the ObjectCache using corba events.

#### **5.6.1.1.3 ProxyDMSPlanItem (Class)**

This class is a proxy for a DMSPlanItem in CHART. It contains a DMSPlanItemData and the stored message id.

#### **5.6.1.1.4 ProxyPlan (Class)**

This class is a proxy object for a Plan in CHART. It contains a list of ProxyPlanItems that it is responsible for keeping updated and providing when needed.

#### **5.6.1.1.5 ProxyPlanItem (Class)**

This class is an abstract base class for specific types of plan items. Ex. ProxyDMSPlanItem.

#### **5.6.1.1.6 QueueableCommand (Class)**

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

### 5.6.1.2 ObjectCacheDMSMsgTemplateRelatedClasses (Class Diagram)

This diagram shows the processing that occurs when the TrafficEventModule object is initialized. The TrafficEventModuleProperties object is created and the frequency (in minutes) to update the nearby devices cache is determined. The TrafficEventDB object is created. The TrafficEventFactoryImpl object is created. The discovery manager is used to update the system profile properties. A command to update object cache with ProxyDMS objects (DiscoverDMSClassesCmd) is created and added to the discovery manager. A command to update object cache with ProxyPlan objects (DiscoverPlanClassesCmd) is created and added to the discovery manager. A command to update object cache with ProxyDMSMsgTemplate objects (DiscoverDMSMsgTemplateClassesCmd) is created and added to the discovery manager. A Timer object is created for use in scheduling timer tasks. A command object for updating the cache with nearby devices information (GetNearbyDeviceInfoCmd) is created and scheduled based on the frequency (in minutes) to update the nearby devices cache.

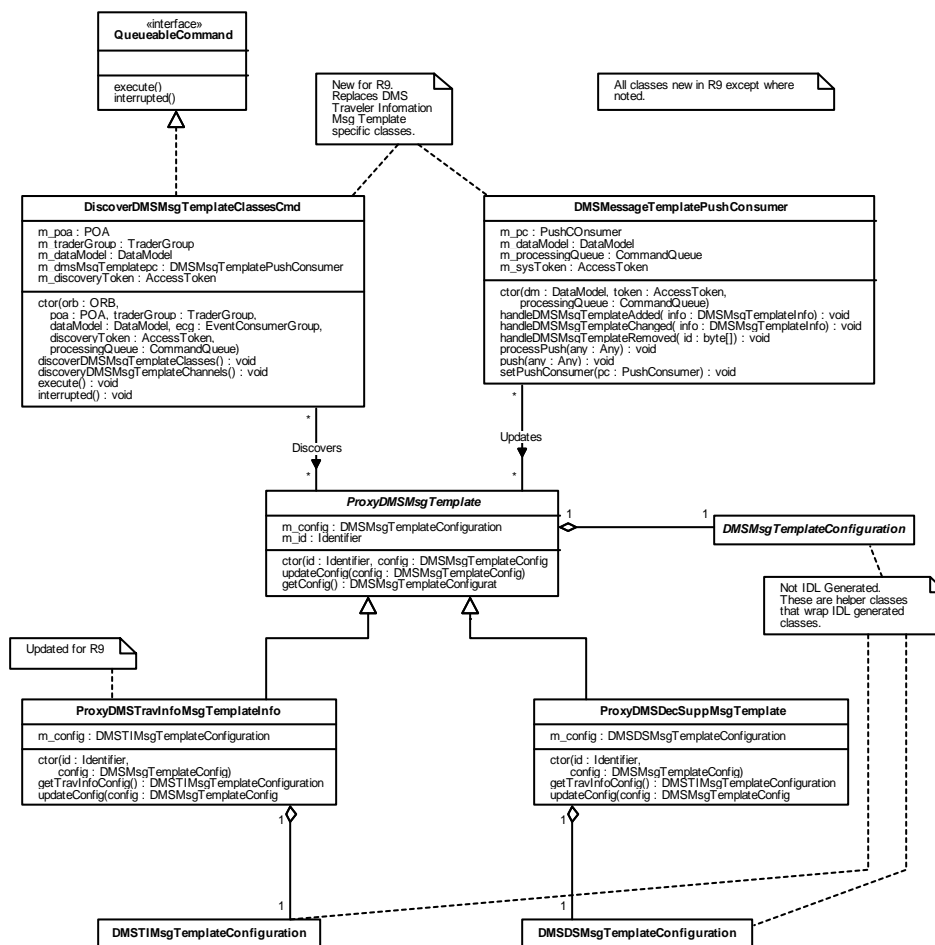


Figure 5-55. ObjectCacheDMSMsgTemplateRelatedClasses (Class Diagram)

#### **5.6.1.2.1 DiscoverDMSMsgTemplateClassesCmd (Class)**

This class is a QueueableCommand that is used by the DiscoveryDriver to maintain local copies of DMS Message Template objects in the object cache. This class contains a PushConsumer that is used to keep data in the object cache up-to-date by handling CORBA events related to DMS Message Templates.

#### **5.6.1.2.2 DMSDMSMsgTemplateConfiguration (Class)**

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSDecSuppMsgTemplate. This is a helper class that provides access to the DMSDecSuppMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

#### **5.6.1.2.3 DMSMessageTemplatePushConsumer (Class)**

This class handles CORBA events pushed by the server on a MessageTemplate event channel. Updates received in events received via the push() method of the PushConsumer interface are updated in the DataModel.

#### **5.6.1.2.4 DMSMsgTemplateConfiguration (Class)**

This abstract base class is extended by classes that represent type specific DMS message template configurations. They are helper classes that make it easier to use hierarchies of IDL structs. These classes wrap IDL generated classes hiding the details of how data is structured in the IDL generated classes.

#### **5.6.1.2.5 DMSTIMsgTemplateConfiguration (Class)**

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSTravInfoMsgTemplate. This is a helper class that provides access to the DMSTravInfoMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

#### **5.6.1.2.6 ProxyDMSDecSuppMsgTemplate (Class)**

This class represents a proxy for a DMS Decision Support Message Template.

#### **5.6.1.2.7 ProxyDMSMsgTemplate (Class)**

This abstract class is the base class for proxy objects that represent specific types of DMS Message Templates.

#### **5.6.1.2.8 ProxyDMSTravInfoMsgTemplateInfo (Class)**

This class represents a proxy for a DMS Traveler Information Message Template.

#### **5.6.1.2.9 QueueableCommand (Class)**

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

## 5.7 MessageTemplateModule

### 5.7.1 Class Diagrams

#### 5.7.1.1 MessageTemplateModule (Class Diagram)

This class diagram shows the classes used by the MessageTemplateModule to provide message template functionality in CHART. Currently two types of message templates are supported, DMS Traveler Information message templates and DMS Decision Support message templates.

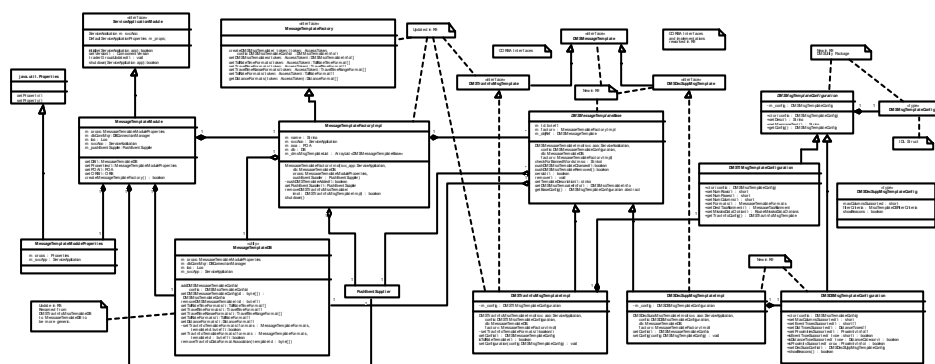


Figure 5-56. MessageTemplateModule (Class Diagram)

##### 5.7.1.1.1 DMSDecSuppMsgTemplate (Class)

The DMSDecSuppMsgTemplate interface is implemented by objects that DMS message templates used for decision support functionality (dms message suggestion generation).

##### 5.7.1.1.2 DMSDecSuppMsgTemplateConfig (Class)

This object contains the configuration data for a message template that represents a DMSDecSuppMsgTemplate in the CHART DB

##### 5.7.1.1.3 DMSDecSuppMsgTemplateImpl (Class)

This class implements the DMSDecSuppMsgTemplate interface defined in the IDL. It provides a set of methods used when working with DMS decision support message templates.

##### 5.7.1.1.4 DMSDSMsgTemplateConfiguration (Class)

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSDecSuppMsgTemplate. This is a helper class that provides access to the DMSDecSuppMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are

structured.

#### **5.7.1.1.5 DMSMessageTemplate (Class)**

This interface is extended by interfaces representing specific types of DMS message templates (ex. DMSTravInfoMsgTemplater). It contains methods common to all types of DMS message templates.

#### **5.7.1.1.6 DMSMessageTemplateBase (Class)**

This abstract class is a base that is extended by DMS message template type specific implementations. It provides a generic class to allow DMS message template to be stored in collections and common functionality like persistence and CORBA event pushing.

#### **5.7.1.1.7 DMSMsgTemplateConfig (Class)**

This struct represents configuration data that is common to all DMS message template types. It contains a union that is used specify DMS message type specific configuration data.

#### **5.7.1.1.8 DMSMsgTemplateConfiguration (Class)**

This abstract base class is extended by classes that represent type specific DMS message template configurations. They are helper classes that make it easier to use hierarchies of IDL structs. These classes wrap IDL generated classes hiding the details of how data is structured in the IDL generated classes.

#### **5.7.1.1.9 DMSTIMsgTemplateConfiguration (Class)**

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSTravInfoMsgTemplate. This is a helper class that provides access to the DMSTravInfoMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

#### **5.7.1.1.10 DMSTravInfoMsgTemplate (Class)**

The DMSTravInfoMsgTemplate interface is implemented by objects that allow execution of tasks associated with DMS travel information message templates.

#### **5.7.1.1.11 DMSTravInfoMsgTemplateImpl (Class)**

This class implements the DMSTravInfoMsgTemplate interface defined in the IDL. It provides a set of methods used when working with DMS travel info message templates.

#### **5.7.1.1.12 java.util.Properties (Class)**

The Properties class represents a persistent set of properties. The Properties can be saved to

a stream or loaded from a stream. Each key and its corresponding value in the property list is a string. A property list can contain another property list as its "defaults"; this second property list is searched if the property key is not found in the original property list.

#### **5.7.1.1.13 MessageTemplateDB (Class)**

This class is a utility that provides methods for adding, removing, and updating database data pertaining to DMS Message Templates.

#### **5.7.1.1.14 MessageTemplateFactory (Class)**

Interface whose implementation is used to create message templates, retrieve message templates and retrieve travel time and toll rate formats.

#### **5.7.1.1.15 MessageTemplateFactoryImpl (Class)**

This class is an implementation of MessageTemplateFactory and is capable of creating a new DMSMsgTemplate objects in the system. Additionally, it provides the services of retrieving existing travel information message formats and DMSMsgTemplates.

#### **5.7.1.1.16 MessageTemplateModule (Class)**

This class provides the resources and support functionality necessary to serve DMSMsgTemplates related objects in a service application. It implements the ServiceApplicationModule interface which allows it to be served from any ServiceApplication.

#### **5.7.1.1.17 MessageTemplateModuleProperties (Class)**

This class provides a wrapper to the application's properties file that provides easy access to the properties specific to the MessageTemplateModule. These properties include the name of the file where raw DMS Message Template parameter data is to be logged, the directory where debug log files are to be kept, and the interval at which the configuration of message formats and DMSTravInfoMsgTemplates is to be read from the DB when requested.

#### **5.7.1.1.18 PushEventSupplier (Class)**

This class provides a utility for application modules that push events on an event channel. The user of this class can pass a reference to the event channel factory to this object. The constructor will create a channel in the factory. The push method is used to push data on the event channel. The push method is able to detect if the event channel or its associated objects have crashed. When this occurs, a flag is set, causing the push method to attempt to reconnect the next time push is called. To avoid a supplier with a heavy supply load from causing reconnect attempts to occur too frequently, a maximum reconnect interval is used. This interval specifies the quickest reconnect interval that can be used. The push method uses this interval and the current time to determine if a reconnect should be attempted, thus reconnects can be throttled independently of a supplier's push rate.



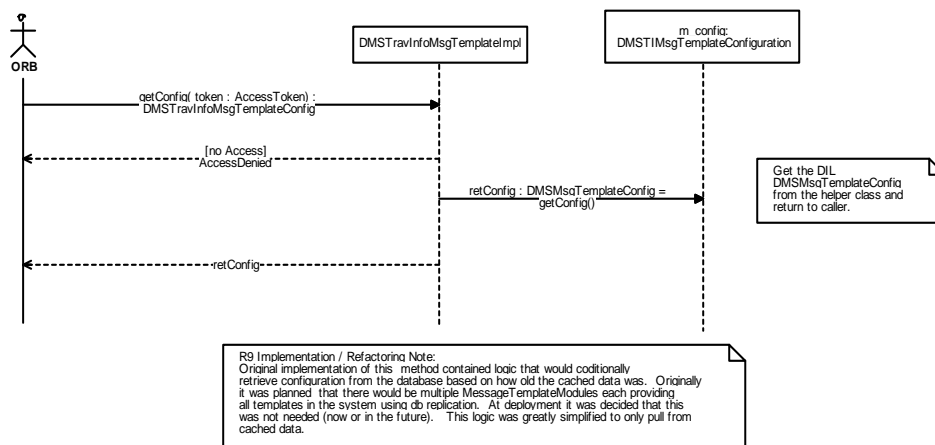
#### **5.7.1.1.19 ServiceApplicationModule (Class)**

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

## 5.7.2 Sequence Diagrams

### 5.7.2.1 DMSTravInfoMsgTemplateImpl:getConfig (Sequence Diagram)

This diagram depicts the DMSTravInfoMsgTemplateImpl.getConfig() method. The function first checks to see if the caller has the correct access rights. If the caller does not have the proper rights the function throws an AccessDenied exception. The method then retrieves the cached copy of the configuration from the impl and returns it to the caller. R9 Implementation / Refactoring Note: Original implementation of this method contained logic that would conditionally retrieve configuration from the database based on how old the cached data was. Originally it was planned that there would be multiple MessageTemplateModules each providing all templates in the system using db replication. At deployment it was decided that this was not needed (now or in the future). This logic was greatly simplified to only pull from cached data.



**Figure 5-57. DMSTravInfoMsgTemplateImpl:getConfig (Sequence Diagram)**

### 5.7.2.2 DMSTravInfoMsgTemplateImpl:setConfig (Sequence Diagram)

This diagram depicts the DMSTravInfoMsgTemplateImpl.setConfig() method. The method first checks the user's access rights. The method then checks that the DMSMsgTemplateConfig pass in is for the correct type of DMSMsgTemplate. If not a CHART2Exception is thrown. Next, the actual message template text is checked for banned words. A CHART2Exception is thrown if banned words are found. The impl's config is then updated. The new config info is then persisted to the database and a DMSMsgTemplateChanged corba event is pushed. Similar processing is done for the DMSDecSupMsgTemplateImpl.

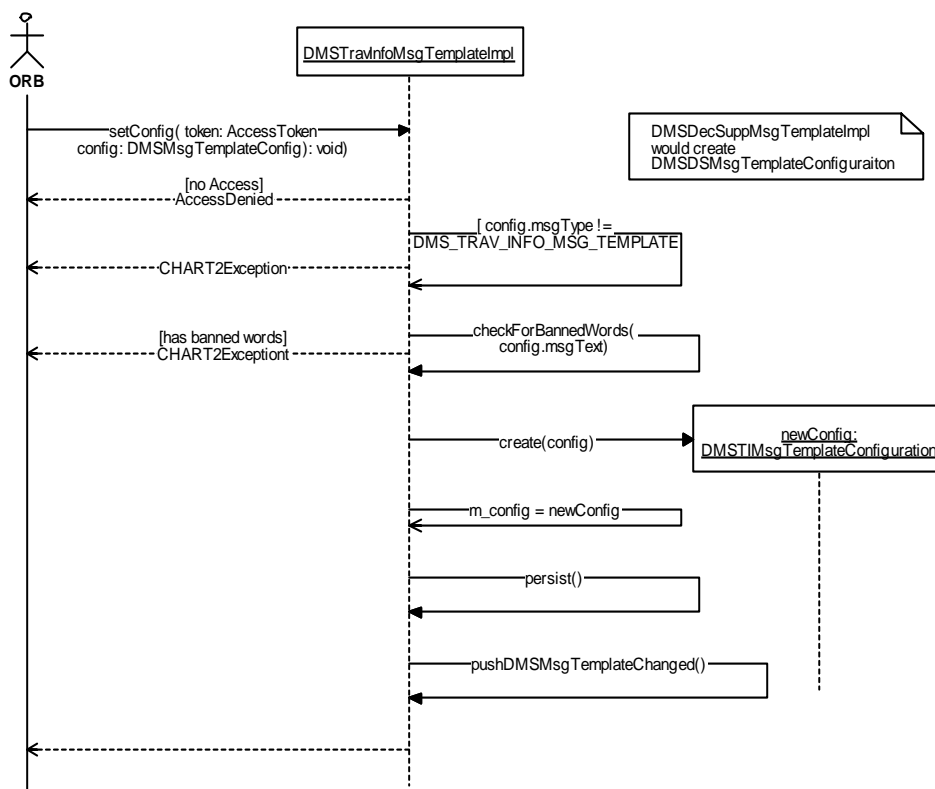


Figure 5-58. DMSTravInfoMsgTemplateImpl:setConfig (Sequence Diagram)

### 5.7.2.3 MessageTemplateFactoryImpl:createDMSMsgTemplate (Sequence Diagram)

This diagram depicts the MessageTemplateFactoryImpl.createDMSMsgTemplate() method. First, the caller's functional rights are checked. An AccessDenied exception is thrown if needed. Next, the appropriate DMSMsgTemplate impl is created based on the supplied config's msgTemplateType value (currently either DMSTravInfoMsgTemplate or DMSDecSuppMsgTemplate). After creation the impl is activated by the POA. A CORBA object reference is created for the template and cached in the templateImpl to be used later when the impl is queried from the factory. Then the template information is persisted to the database, a DMSTemplateAdded corba event is pushed, and the impl is added to the factory's cache. A DMSMsgTemplateInfo object is returned to the caller containing the template's unique id, the template config, and a CORBA reference to the newly create object.

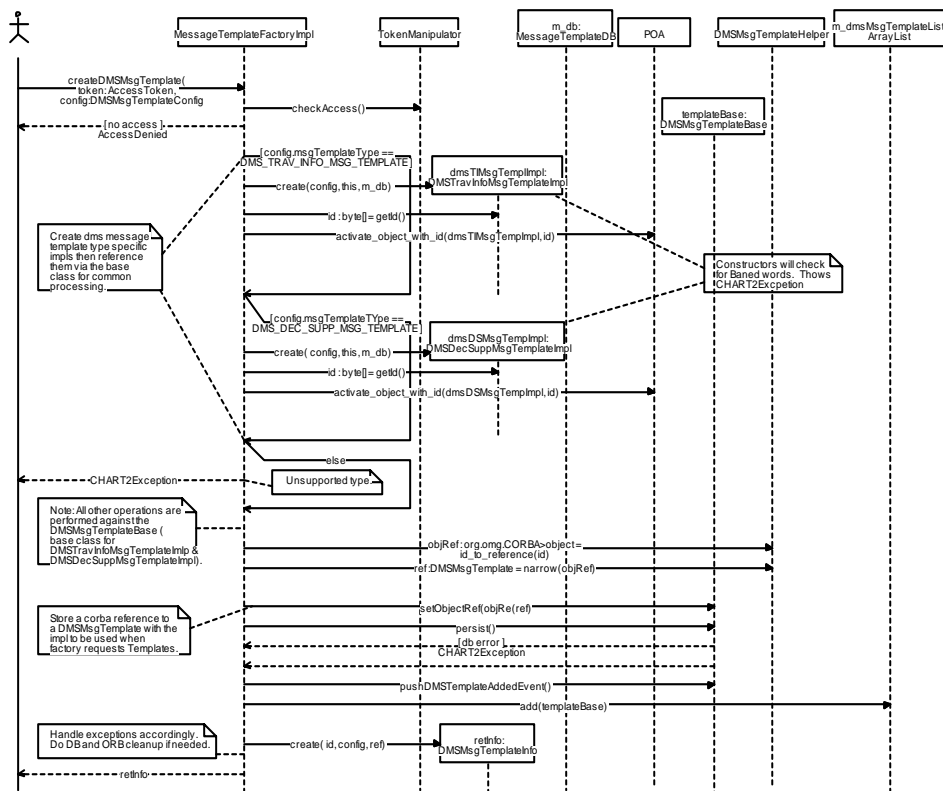
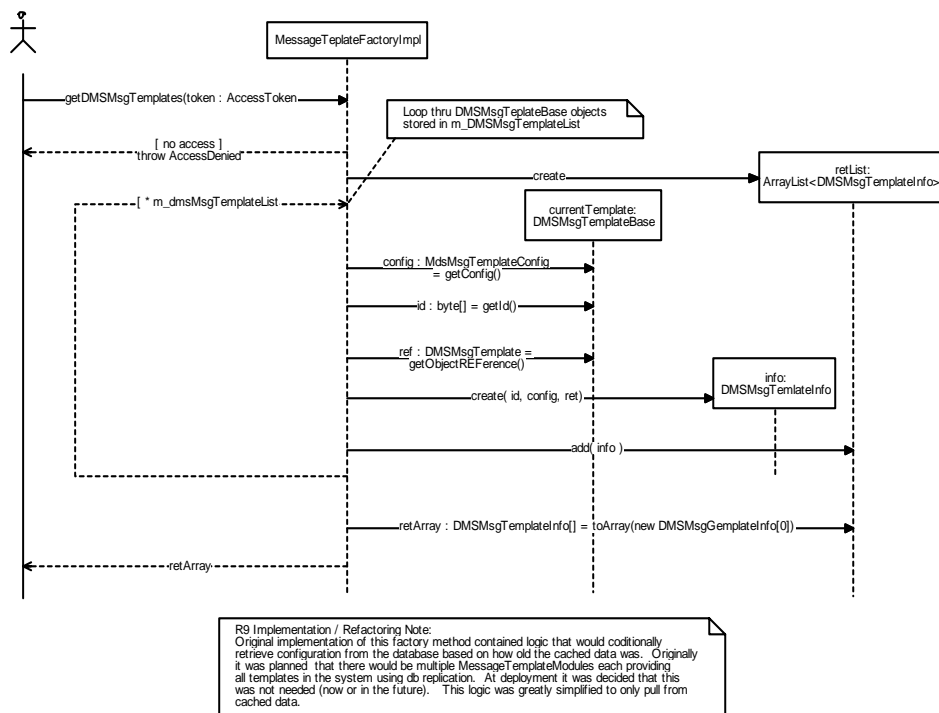


Figure 5-59. MessageTemplateFactoryImpl:createDMSMsgTemplate (Sequence Diagram)

### 5.7.2.4 MessageTemplateFactoryImpl:getDMSMsgTemplates (Sequence Diagram)

This method depicts the getDMSMsgTempaltes() method of the MessageTemplateFactoryImpl. Functional rights are checked to make sure the caller has the appropriate rights. The method then loops through the DMSMsgTemplateImpls currently cached by the factory, creating a DMSMsgTemplateInfo object for each one. These objects are added to a list to be returned to the caller. R9 Implementation / Refactoring Note: Original implementation of this factory method contained logic that would conditionally retrieve configuration from the database based on how old the cached data was. Originally it was planned that there would be multiple MessageTemplateModules each providing all templates in the system using db replication. At deployment it was decided that this was not needed (now or in the future). This logic was greatly simplified to only pull from cached data.



**Figure 5-60. MessageTemplateFactoryImpl:getDMSMsgTemplates (Sequence Diagram)**

## 5.8 DMSUtility (DMS Templates)

### 5.8.1 Class Diagrams

#### 5.8.1.1 DMSMsgTemplateCommonClasses (Class Diagram)

This diagram contains common classes used in formatting the MULTI for a DMSMsgTemplates.

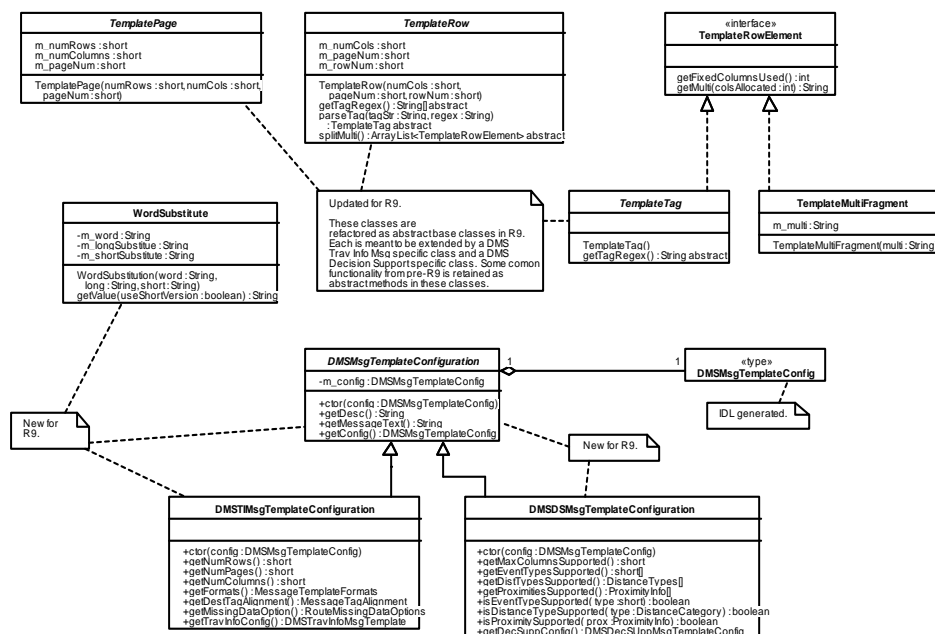


Figure 5-61. DMSMsgTemplateCommonClasses (Class Diagram)

##### 5.8.1.1.1 DMSDecSuppMsgTemplateConfiguration (Class)

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSDecSuppMsgTemplate. This is a helper class that provides access to the DMSDecSuppMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

##### 5.8.1.1.2 DMSMsgTemplateConfig (Class)

This struct represents configuration data that is common to all DMS message template types. It contains a union that is used specify DMS message type specific configuration data.

#### **5.8.1.1.3 DMSMsgTemplateConfiguration (Class)**

This abstract base class is extended by classes that represent type specific DMS message template configurations. They are helper classes that make it easier to use hierarchies of IDL structs. These classes wrap IDL generated classes hiding the details of how data is structured in the IDL generated classes.

#### **5.8.1.1.4 DMSTMsgTemplateConfiguration (Class)**

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSTravInfoMsgTemplate. This is a helper class that provides access to the DMSTravInfoMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

#### **5.8.1.1.5 TemplateMultiFragment (Class)**

This class is the portion of a template row between the data tags (if applicable). The MULTI is kept intact so that MULTI tags such as line justification can be preserved in the output MULTI.

#### **5.8.1.1.6 TemplatePage (Class)**

This class is an abstract base class that represents a page of the template that has been parsed into model form. It is extended by DMS travel information and DMS decision support version with type specific business logic. This class is refactored in R9.

#### **5.8.1.1.7 TemplateRow (Class)**

This class is an abstract base class that represents a row of the template page that has been parsed into model form. It is extended by DMS travel information and DMS Decision Support specific sub classes.

#### **5.8.1.1.8 TemplateRowElement (Class)**

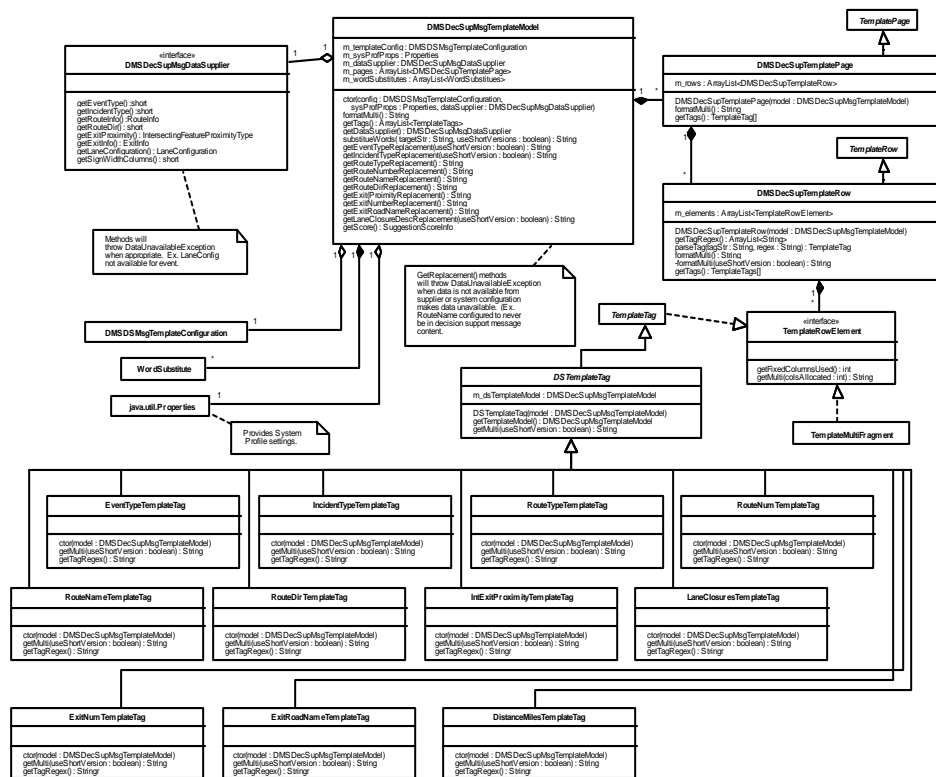
This interface represents an element (component) of the message template row, which can either be a data tag or a fragment of the MULTI outside of the tags.

#### **5.8.1.1.9 TemplateTag (Class)**

This is an abstract base class for template tags. It is extended by DMS Travel Information and DMS Decision Support specific subclasses that provide type specific business rules.

#### **5.8.1.1.10 WordSubstitute (Class)**

The class represents a Word or phrase substitute. It contains the word/phrase to be substituted and a long and short version of the actual substitution string.





used in decision support.

#### **5.8.1.2.3 DMSDecSupTemplatePage (Class)**

This class extends the TemplatePage abstract base class and provides Decision Support business rules DMS Template Pages.

#### **5.8.1.2.4 DMSDecSupTemplateRow (Class)**

This class extends the TemplateRow abstract base class and provides Decision Support business rules DMS Template Rows

#### **5.8.1.2.5 DMSDSMsgTemplateConfiguration (Class)**

This class derives from DMSMsgTemplateConfiguration and represents configuration data for a DMSDecSupMsgTemplate. This is a helper class that provides access to the DMSDecSupMsgTemplateConfig specific data while the base class provides access to the DMSMsgTemplateConfig. It hides the details of how the IDL generated classes are structured.

#### **5.8.1.2.6 DSTemplateTag (Class)**

This class extends the abstract base class TemplateTag and is the base class for specific Decision Support message template tags such as Even Type, Route Type, and Lane Closure.

#### **5.8.1.2.7 EventTypeTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Event Type Tag used in Decision Support Message Templates.

#### **5.8.1.2.8 ExitNumTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Exit Number Tag used in Decision Support Message Templates.

#### **5.8.1.2.9 ExitRoadNameTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Exit Road Name Tag used in Decision Support Message Templates.

#### **5.8.1.2.10 IncidentTypeTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Incident Type Tag used in Decision Support Message Templates.

#### **5.8.1.2.11 IntExitProximityTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Intersecting Exit Proximity Tag

used in Decision Support Message Templates.

#### **5.8.1.2.12 java.util.Properties (Class)**

The Properties class represents a persistent set of properties. The Properties can be saved to a stream or loaded from a stream. Each key and its corresponding value in the property list is a string. A property list can contain another property list as its "defaults"; this second property list is searched if the property key is not found in the original property list.

#### **5.8.1.2.13 LaneClosuresTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Lane Closure Tag used in Decision Support Message Templates.

#### **5.8.1.2.14 RouteDirTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Route Direction Tag used in Decision Support Message Templates.

#### **5.8.1.2.15 RouteNameTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Route Name Tag used in Decision Support Message Templates.

#### **5.8.1.2.16 RouteNumTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Route Number Tag used in Decision Support Message Templates.

#### **5.8.1.2.17 RouteTypeTemplateTag (Class)**

This class extends the DSTemplateTag and represents the Route Type Tag used in Decision Support Message Templates.

#### **5.8.1.2.18 TemplateMultiFragment (Class)**

This class is the portion of a template row between the data tags (if applicable). The MULTI is kept intact so that MULTI tags such as line justification can be preserved in the output MULTI.

#### **5.8.1.2.19 TemplatePage (Class)**

This class is an abstract base class that represents a page of the template that has been parsed into model form. It is extended by DMS travel information and DMS decision support version with type specific business logic. This class is refactored in R9.

#### **5.8.1.2.20 TemplateRow (Class)**

This class is an abstract base class that represents a row of the template page that has been

parsed into model form. It is extended by DMS travel information and DMS Decision Support specific sub classes.

#### **5.8.1.2.21 TemplateRowElement (Class)**

This interface represents an element (component) of the message template row, which can either be a data tag or a fragment of the MULTI outside of the tags.

#### **5.8.1.2.22 TemplateTag (Class)**

This is an abstract base class for template tags. It is extended by DMS Travel Information and DMS Decision Support specific subclasses that provide type specific business rules.

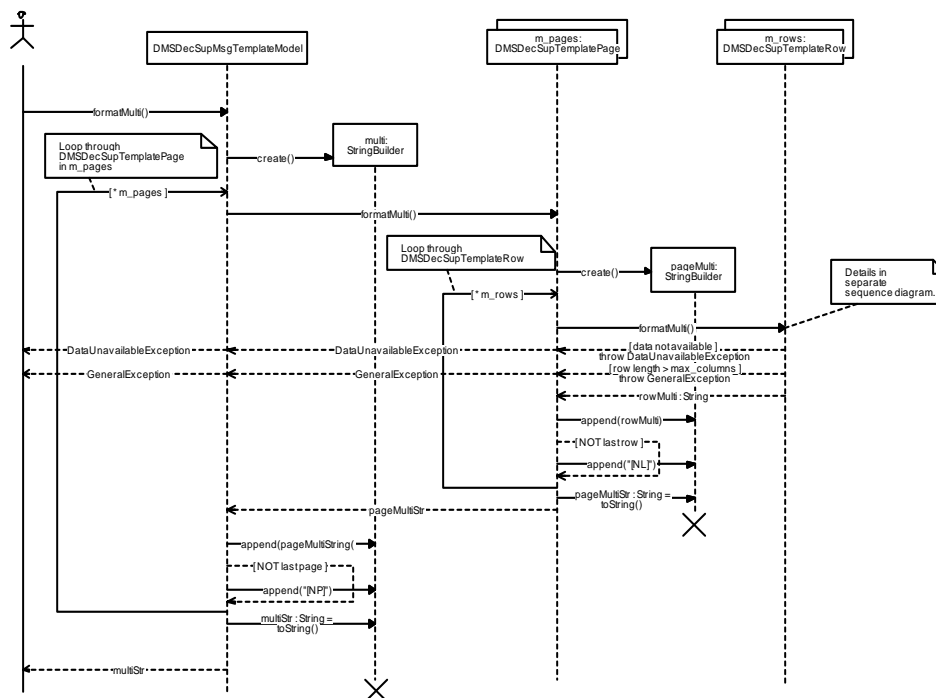
#### **5.8.1.2.23 WordSubstitute (Class)**

The class represents a Word or phrase substitute. It contains the word/phrase to be substituted and a long and short version of the actual substitution string.

## 5.8.2 Sequence Diagrams

### 5.8.2.1 DMSDecSupMsgTemplateModel:formatMulti (Sequence Diagram)

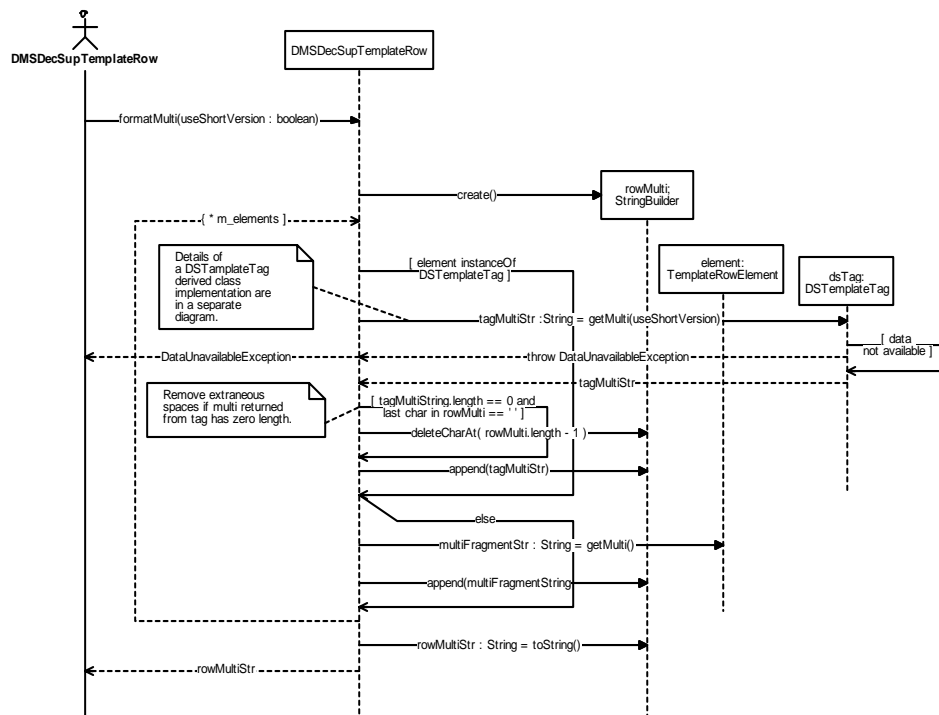
This diagram depicts processing done when formatting a decision support message suggestion. The Model is created with a DMS Decision Support data supplier that will supply data for the specific DMS and Event. When the formatMulti() method is called the model loops through its existing DMSDecSupTemplatePage objects calling furmatMulti() on each one. For each page the returned formatted string will have all decision support tags replaced with values supplied by the supporter and model. Those strings are used to create a multi message string that is returned to the caller. Exceptions are thrown If the data cannot be supplied for a specific tag or a row exceeds the width of the sign in columns after tag replacement. If that occurs, no message will be formation for the template.



**Figure 5-63. DMSDecSupMsgTemplateModel:formatMulti (Sequence Diagram)**

### 5.8.2.2 DMSDecSupTemplateRow:formatMultiPrivate (Sequence Diagram)

This diagram depicts the DMSDecSupTemplateRow.FormatMulti() method that is called specifically for long or short replacement values. This method loops through the TemplateRowElements in a row building the actual multi for the row using tag replace values. For elements of type DSTemplateTag, it will do tag replacement with either the short or long version based on the useShortVersion boolean param. For elements of type MultiFragment just the multi fragment string is used. If data can be supplied for all tags in the row, the multi string for the row is returned.



**Figure 5-64. DMSDecSupTemplateRow:formatMultiPrivate (Sequence Diagram)**

### 5.8.2.3 DMSDecSupMsgTemplateRow:formatMultiPublic (Sequence Diagram)

This diagram depicts the DMSDecSupMsgTemplateRow.formatMulti() method. First, an attempt is made to replace tags on the row using long versions of replacement values if available. If that attempt produces a row that is too long for the width of the sign, tag replacement is attempted using short versions of values if available. If that attempt fails to produce a row that will fit on the sign, a GeneralException is thrown. At any time, if data cannot be supplied for a specific tag, a DataUnavailableException is thrown.

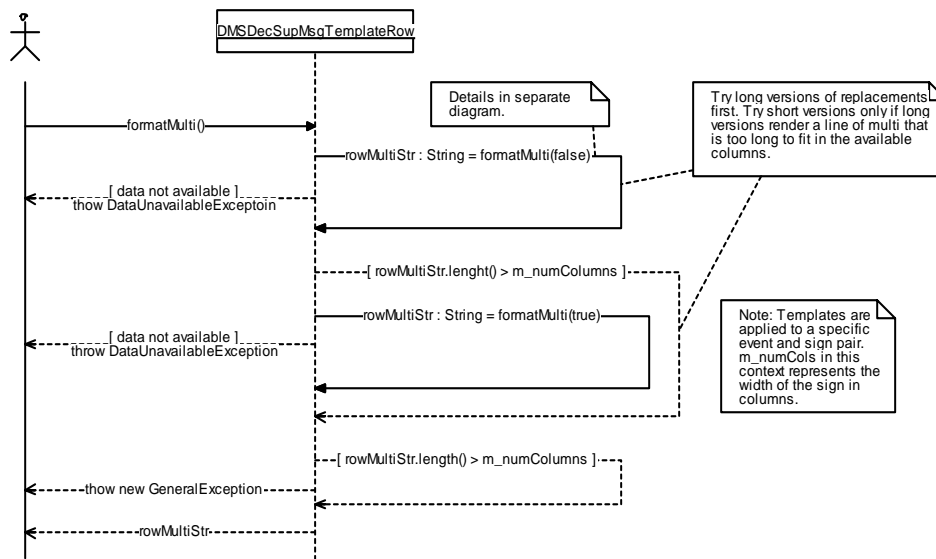


Figure 5-65. DMSDecSupMsgTemplateRow:formatMultiPublic (Sequence Diagram)

### 5.8.2.4 DSTemplateTag:getMulti (Sequence Diagram)

This diagram depicts examples of the getMulti() method for different decision support tag types. The model is used to provide data for each tag. If the tag supports a long and short replacement value, the useShortVersion param is used, otherwise it is ignored. The model will do any appropriate tag replacement (using long/short values if applicable). When appropriate the resulting tag replacement will be checked for word substitutions that were previously configured in the system.

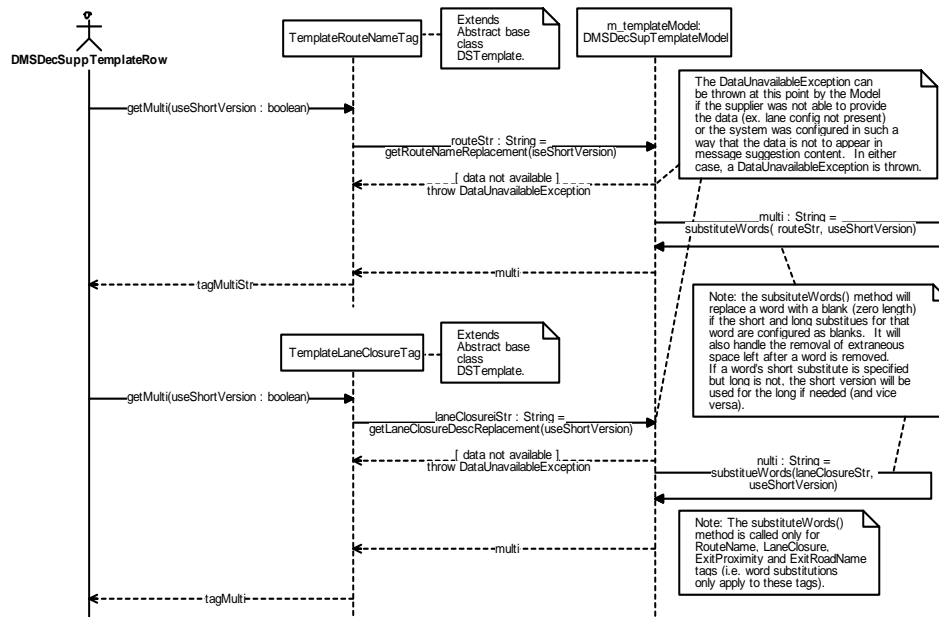


Figure 5-66. DSTemplateTag:getMulti (Sequence Diagram)

## 5.9 chartlite.data.trafficevents

### 5.9.1 Class Diagrams

#### 5.9.1.1 chartlite.data.trafficevents.DecisionSupport (Class Diagram)

This diagram shows GUI classes related to decision support functionality including DMS, plan, and DMS plan item suggestions.

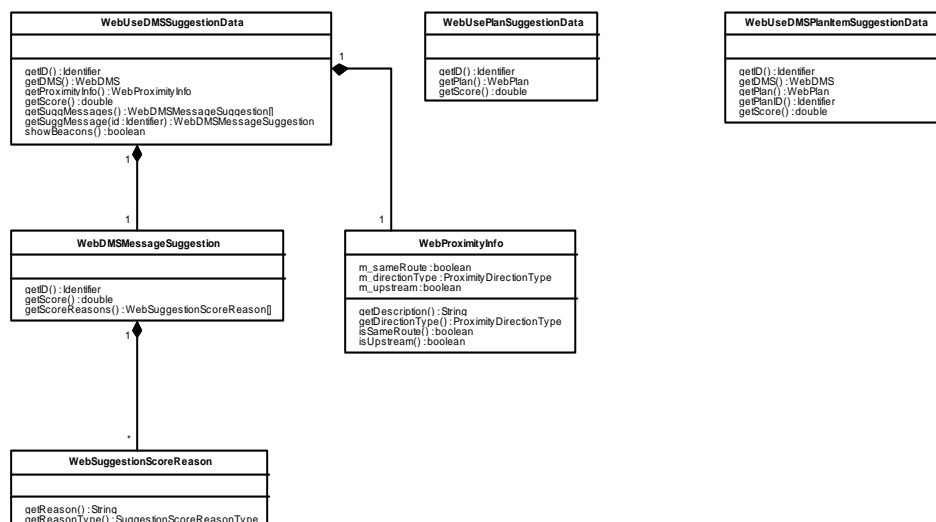


Figure 5-67. chartlite.data.trafficevents.DecisionSupport (Class Diagram)

##### 5.9.1.1.1 WebDMSMessageSuggestion (Class)

This class wraps the DMSMessageSuggestion CORBA structure representing a decision support DMS message suggestion. This class provides accessor methods for use in the GUI.

##### 5.9.1.1.2 WebProximityInfo (Class)

This class wraps the ProximityInfo CORBA structure representing a decision support proximity. This class provides accessor methods for use in the GUI.

##### 5.9.1.1.3 WebSuggestionScoreReason (Class)

This class wraps the SuggestionScoreReason CORBA structure representing a decision support DMS message suggestion score/reason. Both scores and reasons are contained in this object. This class provides accessor methods for use in the GUI.

##### 5.9.1.1.4 WebUseDMSPlanItemSuggestionData (Class)

This class wraps the UseDMSPlanItemSuggestionData CORBA structure representing a



decision support DMS plan item suggestion. This class provides accessor methods for use in the GUI.

#### **5.9.1.1.5 WebUseDMSSuggestionData (Class)**

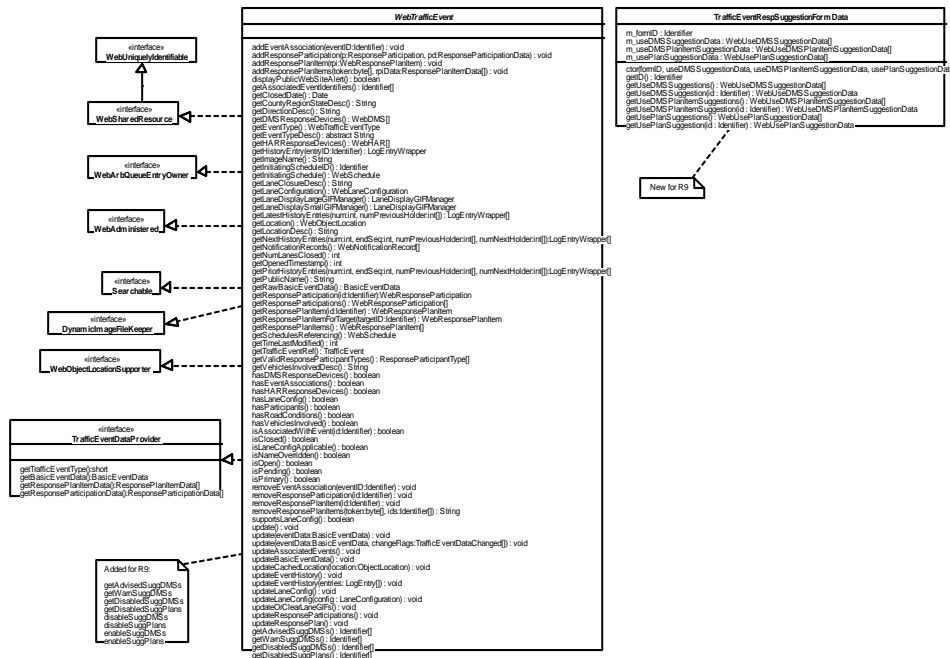
This class wraps the UseDMSSuggestionData CORBA structure representing a decision support DMS suggestion. This class provides accessor methods for use in the GUI.

#### **5.9.1.1.6 WebUsePlanSuggestionData (Class)**

This class wraps the UsePlanSuggestionData CORBA structure representing a decision support plan suggestion. This class provides accessor methods for use in the GUI.

### 5.9.1.2 chartlite.data.trafficevents\_classes (Class Diagram)

This diagram shows the main wrapper class used for storing traffic event related data in the cache, as well as the class required to display decision support suggestions to a user.



**Figure 5-68. chartlite.data.trafficevents\_classes (Class Diagram)**

#### 5.9.1.2.1 DynamicImageFileKeeper (Class)

This interface allows an object to keep dynamic image files from being deleted by the `DynImageCleanupTask`, which periodically deletes files that are no longer needed.

#### 5.9.1.2.2 Searchable (Class)

This interface allows objects to be searched for via a substring search.

#### 5.9.1.2.3 TrafficEventDataProvider (Class)

This interface is implemented by classes that can provide data from a traffic event. This interface exists because traffic event data may be accessible in different forms depending on where the ActionExecutionGroup (and related classes) are being used. For example, the data for a traffic event may be accessible via a cache of traffic event data or via a CORBA object reference.

#### **5.9.1.2.4 TrafficEventRespSuggestionFormData (Class)**

This class represents the data required to display decision support suggestions (DMS, plan, and DMS plan item suggestions) to a user. It provides accessor methods to get the suggestion data.

#### **5.9.1.2.5 WebAdministered (Class)**

This interface allows the implementing class to be administered via the trader console pages.

#### **5.9.1.2.6 WebArbQueueEntryOwner (Class)**

This interface specifies methods to be implemented by all objects that may place entries on an arbitration queue.

#### **5.9.1.2.7 WebObjectLocationSupporter (Class)**

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

#### **5.9.1.2.8 WebSharedResource (Class)**

This interface is implemented by any GUI-side wrapper objects representing CHART shared resources in the system, corresponding to the SharedResource IDL interface.

#### **5.9.1.2.9 WebTrafficEvent (Class)**

This class represents a TrafficEvent object in the system and caches its data for fast access. It provides accessor methods to get the cached data, in addition to auxiliary methods.

#### **5.9.1.2.10 WebUniquelyIdentifiable (Class)**

This interface provides functionality for GUI objects that represent UniquelyIdentifiable objects as defined in the IDL.

### 5.10.1.1 GUIMessageTemplateServletClasses (Class Diagram)

```

classDiagram
    class RequestHandler {
        <<interface>>
        initiateRequest(handler: RequestHandler, request: RequestHandler, context: Context, requestHandler: RequestHandler) String
        processRequest(handler: RequestHandler, request: RequestHandler, context: Context, requestHandler: RequestHandler) String
    }

    class MessageTemplate {
        <<interface>>
        m_dysListDelegate: DysListRequestDelegate
    }

    class DMSRequestData {
        m_addressID: String
        m_subFormID: String
        m_geometry: DMSDisplayInfo
        m_isMenuA: Boolean
        m_isMenuADisabled: Boolean
        m_isReason: Boolean
        m_isReasonEnabled: Boolean
        m_multiLang: String
        m_planetLang: String
        m_langID: String
        m_centerGeometryID: String
        m_messageID: Boolean
        m_needSplash: Boolean
        m_needSplash: String
        m_needCommonGeometry: Boolean
    }

    class DMSResponseData {
        m_template: WebDMSTemplate
        m_dysListDelegate: DysListRequestDelegate
        m_isMenuA: Boolean
        m_isMenuADisabled: Boolean
        m_isReason: Boolean
        m_isReasonEnabled: Boolean
        m_multiLang: String
        m_planetLang: String
        m_langID: String
        m_centerGeometryID: String
        m_messageID: Boolean
        m_needSplash: Boolean
        m_needSplash: String
        m_needCommonGeometry: Boolean
    }

    class DMSRequestTemplate {
        m_template: WebDMSTemplate
        m_dysListDelegate: DysListRequestDelegate
        m_isMenuA: Boolean
        m_isMenuADisabled: Boolean
        m_isReason: Boolean
        m_isReasonEnabled: Boolean
        m_multiLang: String
        m_planetLang: String
        m_langID: String
        m_centerGeometryID: String
        m_messageID: Boolean
        m_needSplash: Boolean
        m_needSplash: String
        m_needCommonGeometry: Boolean
    }

    class DMSResponseTemplate {
        m_template: WebDMSTemplate
        m_dysListDelegate: DysListRequestDelegate
        m_isMenuA: Boolean
        m_isMenuADisabled: Boolean
        m_isReason: Boolean
        m_isReasonEnabled: Boolean
        m_multiLang: String
        m_planetLang: String
        m_langID: String
        m_centerGeometryID: String
        m_messageID: Boolean
        m_needSplash: Boolean
        m_needSplash: String
        m_needCommonGeometry: Boolean
    }

    RequestHandler <|-- DMSRequestData
    RequestHandler <|-- DMSResponseData
    MessageTemplate <|-- DMSRequestTemplate
    MessageTemplate <|-- DMSResponseTemplate
  
```

**NOTE:** The address template functionality is handled in the `handleMenuADisabled` method (DMSRequestData) to take advantage of existing editor code.

**New for R9:**

- `getDMSRequestTemplateList`
- `getDMSRequestTemplate`
- `removeDMSRequestTemplate`

**DMSRequestData**

- `m_addressID: String`
- `m_subFormID: String`
- `m_geometry: DMSDisplayInfo`
- `m_isMenuA: Boolean`
- `m_isMenuADisabled: Boolean`
- `m_isReason: Boolean`
- `m_isReasonEnabled: Boolean`
- `m_multiLang: String`
- `m_planetLang: String`
- `m_langID: String`
- `m_centerGeometryID: String`
- `m_messageID: Boolean`
- `m_needSplash: Boolean`
- `m_needSplash: String`
- `m_needCommonGeometry: Boolean`

**DMSResponseData**

- `m_template: WebDMSTemplate`
- `m_dysListDelegate: DysListRequestDelegate`
- `m_isMenuA: Boolean`
- `m_isMenuADisabled: Boolean`
- `m_isReason: Boolean`
- `m_isReasonEnabled: Boolean`
- `m_multiLang: String`
- `m_planetLang: String`
- `m_langID: String`
- `m_centerGeometryID: String`
- `m_messageID: Boolean`
- `m_needSplash: Boolean`
- `m_needSplash: String`
- `m_needCommonGeometry: Boolean`

**DMSRequestTemplate**

- `m_template: WebDMSTemplate`
- `m_dysListDelegate: DysListRequestDelegate`
- `m_isMenuA: Boolean`
- `m_isMenuADisabled: Boolean`
- `m_isReason: Boolean`
- `m_isReasonEnabled: Boolean`
- `m_multiLang: String`
- `m_planetLang: String`
- `m_langID: String`
- `m_centerGeometryID: String`
- `m_messageID: Boolean`
- `m_needSplash: Boolean`
- `m_needSplash: String`
- `m_needCommonGeometry: Boolean`

**DMSResponseTemplate**

- `m_template: WebDMSTemplate`
- `m_dysListDelegate: DysListRequestDelegate`
- `m_isMenuA: Boolean`
- `m_isMenuADisabled: Boolean`
- `m_isReason: Boolean`
- `m_isReasonEnabled: Boolean`
- `m_multiLang: String`
- `m_planetLang: String`
- `m_langID: String`
- `m_centerGeometryID: String`
- `m_messageID: Boolean`
- `m_needSplash: Boolean`
- `m_needSplash: String`
- `m_needCommonGeometry: Boolean`

**Figure 5-69. GUIMessageTemplateServletClasses (Class Diagram)**

#### 5.10.1.1.1 DMSDecSuppMsgTemplateEditorData (Class)

This class contains the data related to one instance of the DMS decision support message template editor. It extends `DMSEditorData` to provide common editor functionality, and also has methods for getting and setting the editor data.

#### **5.10.1.1.2 DMSEditorData (Class)**

This class represents an instance of a DMS message being edited in an editor. It provides storage so that the message and editor state can be preserved during interim requests before the form is submitted. It also has logic for manipulating the editor session. This is a base class and will be extended for specific editor types.

#### **5.10.1.1.3 DMSTravInfoMsgTemplateEditorData (Class)**

This class contains the data related to one instance of the DMS traveler information message template editor. It extends DMSEditorData to provide common editor functionality, and also has methods for getting and setting the editor data.

#### **5.10.1.1.4 MessageTemplateReqHdlr (Class)**

This class handles requests related to message templates, except for the requests for adding / editing a template, which are in the DMSReqHdlr class.

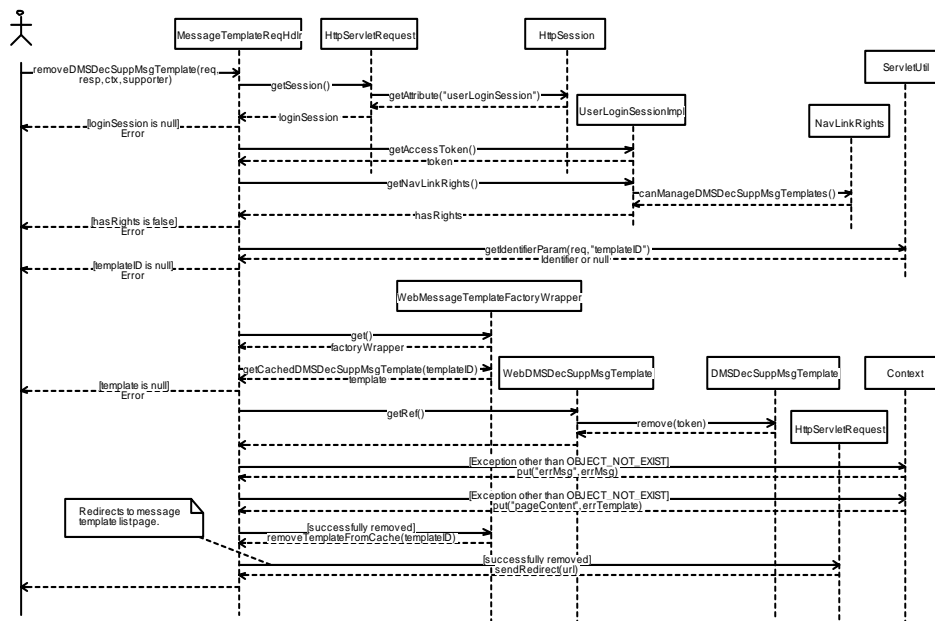
#### **5.10.1.1.5 RequestHandler (Class)**

This interface specifies methods that are to be implemented by classes that are used to process requests.

## 5.10.2 Sequence Diagrams

### 5.10.2.1 MessageTemplateReqHdlr:removeDMSDecSuppMsgTemplate (Sequence Diagram)

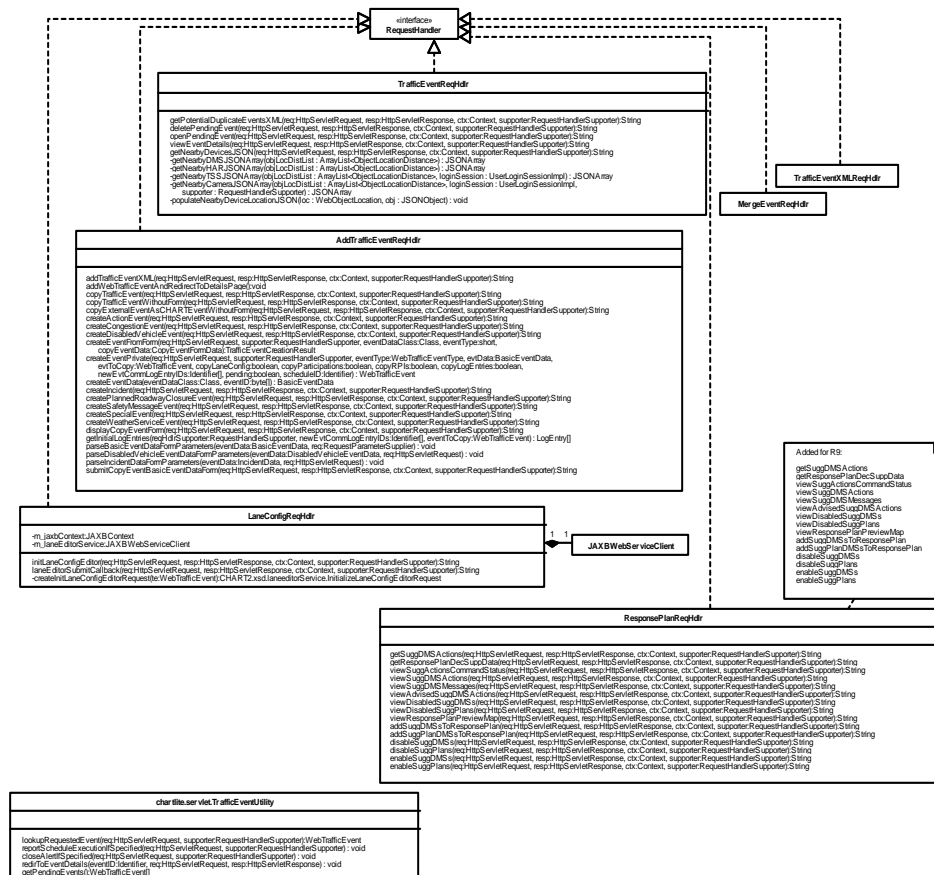
This diagram shows the processing that occurs to remove a DMS decision support message template from the system. The login session is checked to verify the user has the functional rights to remove a message template. The templateID parameter is used to get the WebDMSDecSuppMsgTemplate object from the factory wrapper cache. The DMSDecSuppMsgTemplate CORBA object is called to remove itself. If successful (or an OBJECT\_NOT\_EXIST error occurs), the template is removed from the factory wrapper cache and a response is sent to redirect the browser to the updated list of templates.



**Figure 5-70. MessageTemplateReqHdlr:removeDMSDecSuppMsgTemplate (Sequence Diagram)**

#### 5.11.1.1 chartlite.servlet.trafficevents\_classes (Class Diagram)

This diagram shows the various classes that are used to handle requests related to traffic events.



**Figure 5-71. chartlite.servlet.trafficevents\_classes (Class Diagram)**

#### 5.11.1.1.1 AddTrafficEventReqHdlr (Class)

This class is used to handle requests related to adding a traffic event to the system.

#### 5.11.1.1.2 chartlite.servlet.TrafficEventUtility (Class)

This class contains methods that are useful for one or more traffic event related request handlers.

#### **5.11.1.1.3 JAXBWebServiceClient (Class)**

This class is a wrapper for an XMLHTTPService that makes it easy to send/receive data to/from the service using objects that are generated via xsd and JAXB.

#### **5.11.1.1.4 LaneConfigReqHdlr (Class)**

This class handles any requests related to the traffic event lane configuration. The lane configuration editing has been moved to a web service as of R6, therefore the functionality of this class is mainly to initialize an editing session within the lane editor web service and to handle the results of the lane editing session when called back from the lane editor web service.

#### **5.11.1.1.5 MergeEventReqHdlr (Class)**

This class handles all requests related to merging traffic events.

#### **5.11.1.1.6 RequestHandler (Class)**

This interface specifies methods that are to be implemented by classes that are used to process requests.

#### **5.11.1.1.7 ResponsePlanReqHdlr (Class)**

This class handles requests related to traffic event response plans including decision support suggestions.

#### **5.11.1.1.8 TrafficEventReqHdlr (Class)**

This class handles requests related to traffic events that are not handled by one of the other specific traffic event request handlers.

#### **5.11.1.1.9 TrafficEventXMLReqHdlr (Class)**

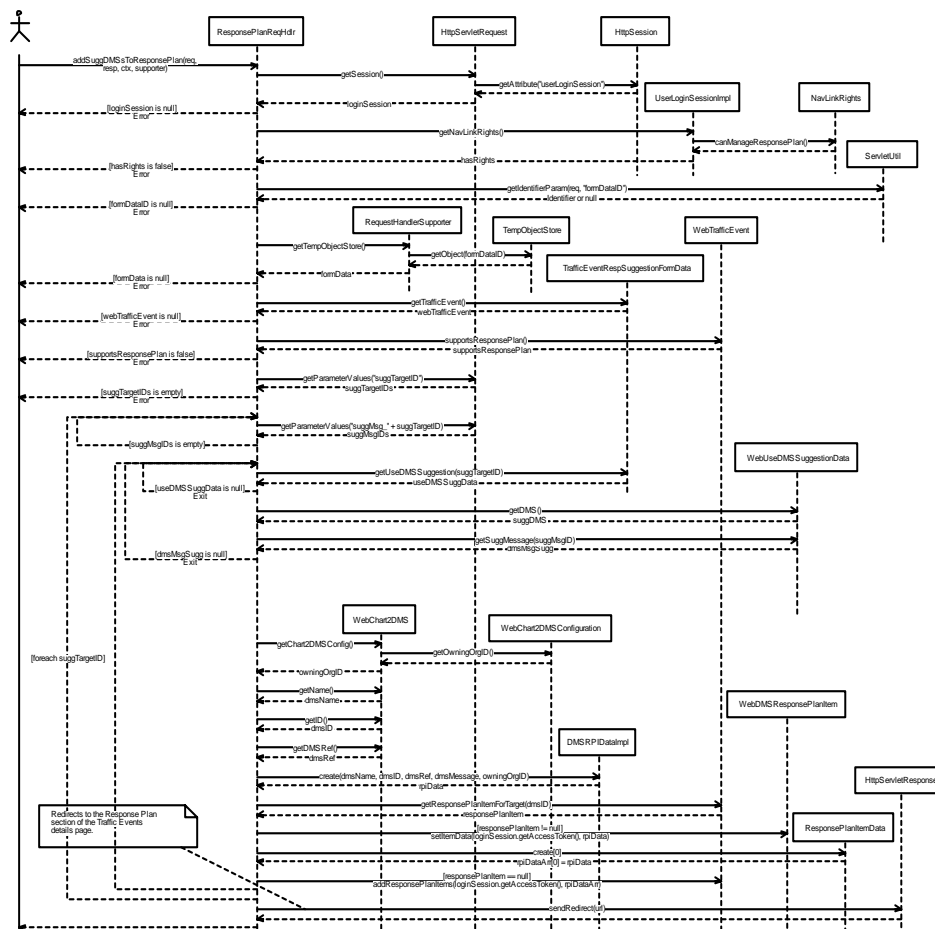
This class handles requests related to traffic events that return XML for the Flex2 application.



## 5.11.2 Sequence Diagrams

### 5.11.2.1 chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:addSuggDMSsToResponsePlan (Sequence Diagram)

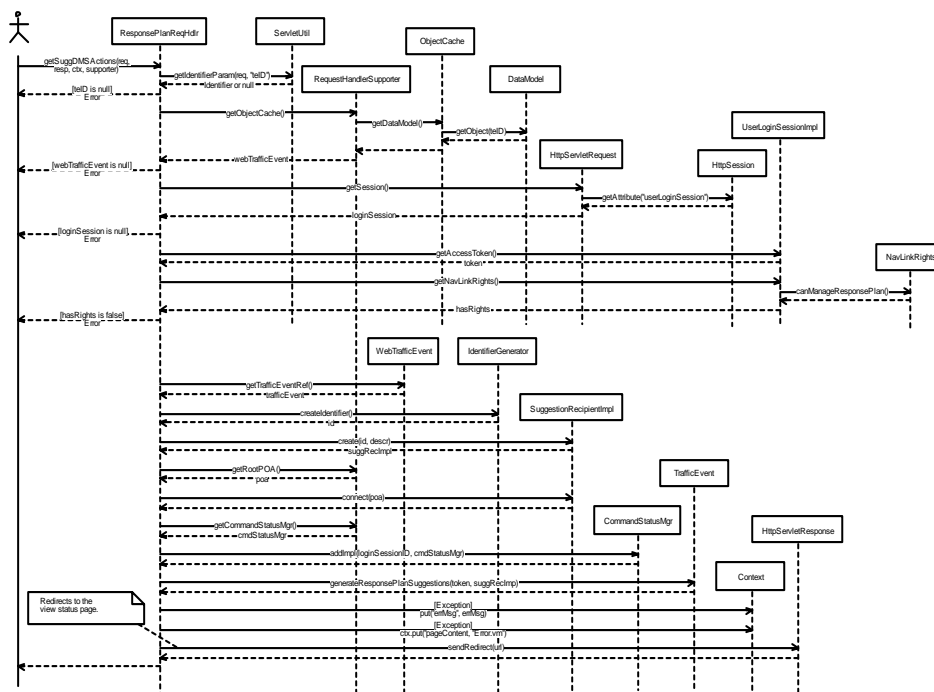
This diagram shows the processing that occurs when the decision support suggestion form is submitted to add a suggestion to the response plan. The login session is checked to verify the user has the functional rights to manage the response plan. The formDataID parameter is used to get the SuggestionFormData object from the temporary object store. The traffic event object is obtained from the suggestion form data and checked to verify it supports a response plan. The suggTargetID and suggMsg parameters are used to determine which DMS/message combinations were selected by the user. For each selected DMS/message combination, a response plan item is created and added to the response plan. A response is sent to redirect the browser to the response plan section of the traffic even details page.



**Figure 5-72.**  
chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:addSuggDMSsToResponsePlan  
(Sequence Diagram)

### 5.11.2.2 chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:getSuggDMSActions (Sequence Diagram)

This diagram shows the processing that occurs when a user requests decision support suggestions for a traffic event. The trafficEventID parameter is used to get the WebTrafficEvent object from the data model. The login session is checked to verify the user has the functional rights to manage the response plan. A reference to the traffic event CORBA object is obtained from the WebTrafficEvent object. A SuggestionRecipientImpl object is created and connected to the root Portable Object Adapter (POA). The SuggestionRecipientImpl object is passed to the traffic event object using the generateResponsePlanSuggestions method. If no errors occur, a response is sent to redirect the browser to the command status page.



**Figure 5-73. chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:getSuggDMSActions  
(Sequence Diagram)**

### 5.11.2.3 chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:viewSuggActionsCommandStatus (Sequence Diagram)

This diagram shows the processing that occurs during the time the traffic event services is updating a SuggestionRecipientImpl with suggestion data. The login session is checked to verify the user does not have view only rights. The cmdStatusID parameter is used to get the CommandStatusImpl object from the command status manager. If the command status has completed, a response is sent to redirect the browser to the decision support suggestions page; otherwise, the context is updated with a refresh interval and the browser is directed back to the command status page.

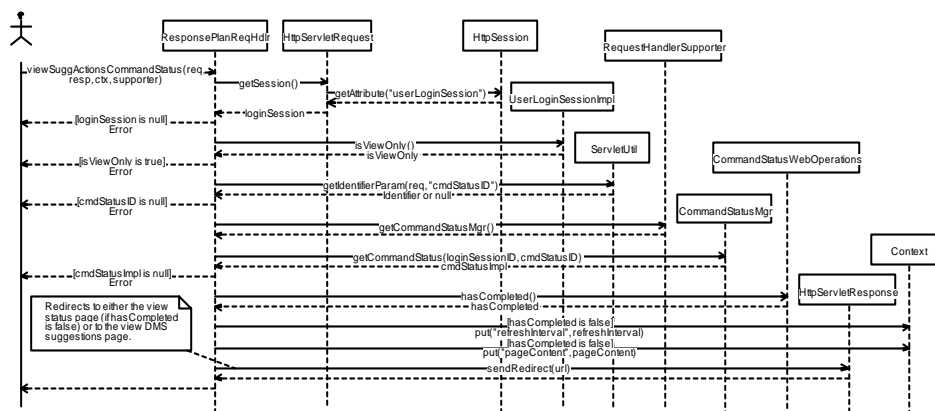
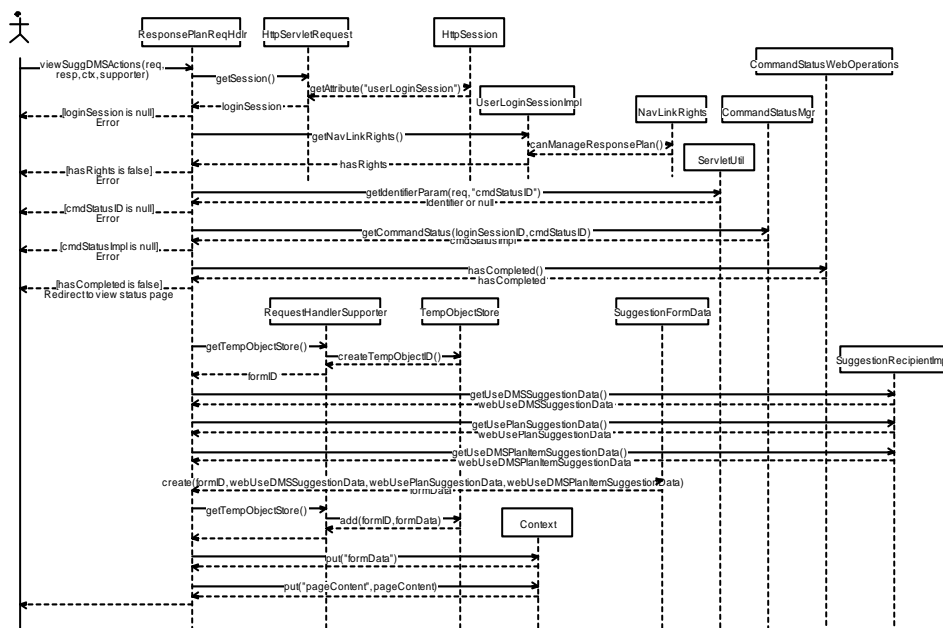


Figure 5-74.

### chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:viewSuggActionsCommandStatus (Sequence Diagram)

#### 5.11.2.4 chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:viewSuggDMSActions (Sequence Diagram)

This diagram shows the processing that occurs when decision support suggestions for a traffic event are displayed to the user. The login session is checked to verify the user has the functional rights to manage the response plan. The cmdStatusID parameter is used to get the CommandStatusImpl object from the command status manager. The CommandStatusImpl object is checked to verify processing has completed. The suggestion data is retrieved from the SuggestionRecipientImpl object. A SuggestionFormData object is created, populated with the suggestion data, added to the temporary object store, and added to the context. The browser is directed to the decision support suggestions page to display the requested suggestions.



**Figure 5-75. chartlite.servlet.trafficEvents.ResponsePlanReqHdlr:viewSuggDMSActions (Sequence Diagram)**

### 5.11.2.5 ResponsePlanReqHdlr:getResponsePlanDecSuppData (Sequence Diagram)

This diagram shows how the list for the decision support data for the recommended and not recommended DMSs is obtained. If the call to `lookupRequestedEvent` fails to locate the `WebTrafficEvent` from the request, an error is returned. Otherwise, a call to the `hasAdvisedSuggDMSActions` and `hasWarnSuggDMSActions` retrieves the suggested and not suggested DMSs. From these lists, those DMSs that are not on the current response plan are added to either the `recommendedDMSList` or `notRecommendedList` respectively. These lists are added to the context along with the current response plan item info.

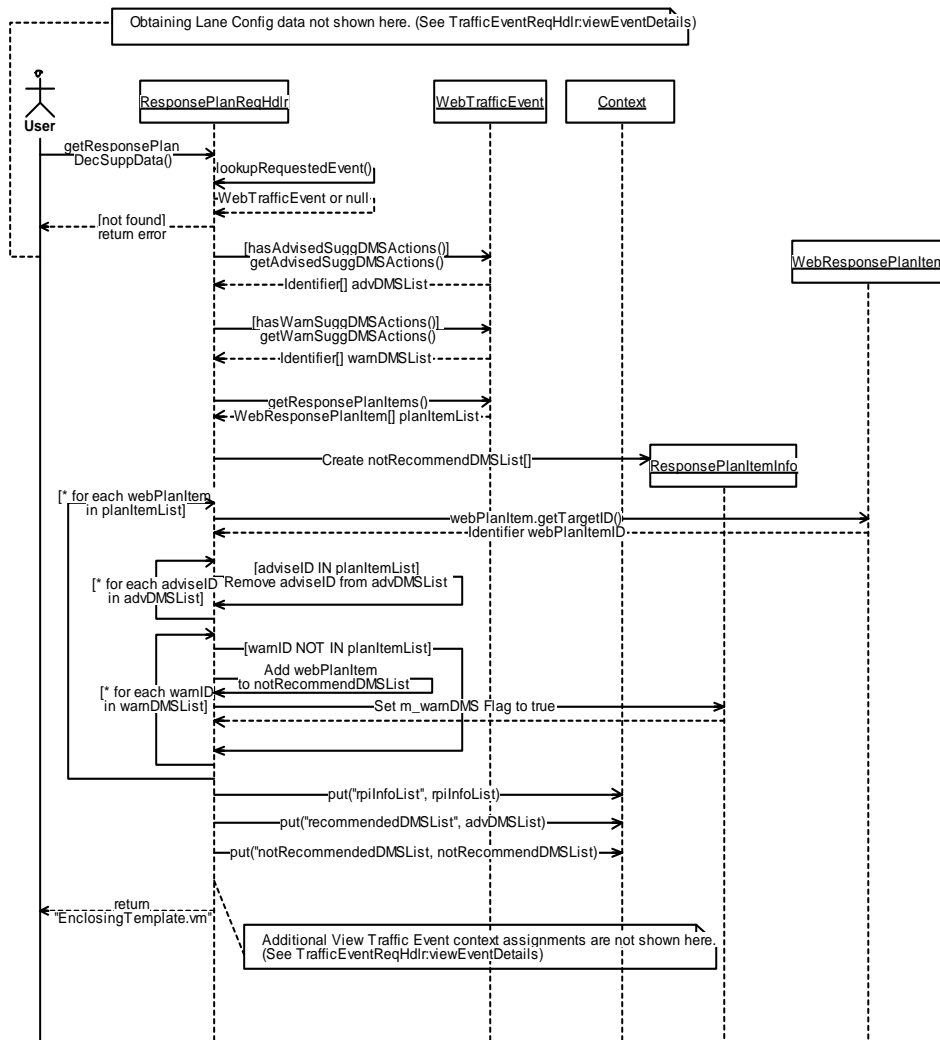
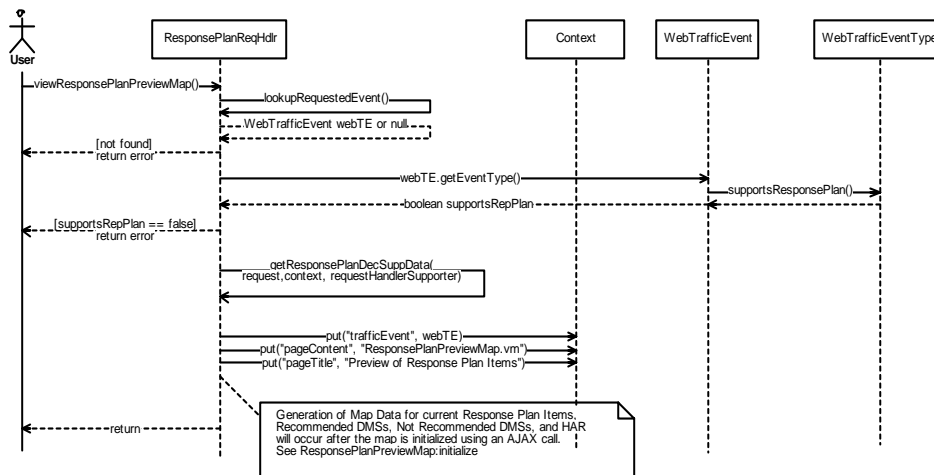


Figure 5-76. ResponsePlanReqHdlr:getResponsePlanDecSuppData (Sequence Diagram)

### 5.11.2.6 ResponsePlanReqHdlr:viewResponsePlanPreviewMap (Sequence Diagram)

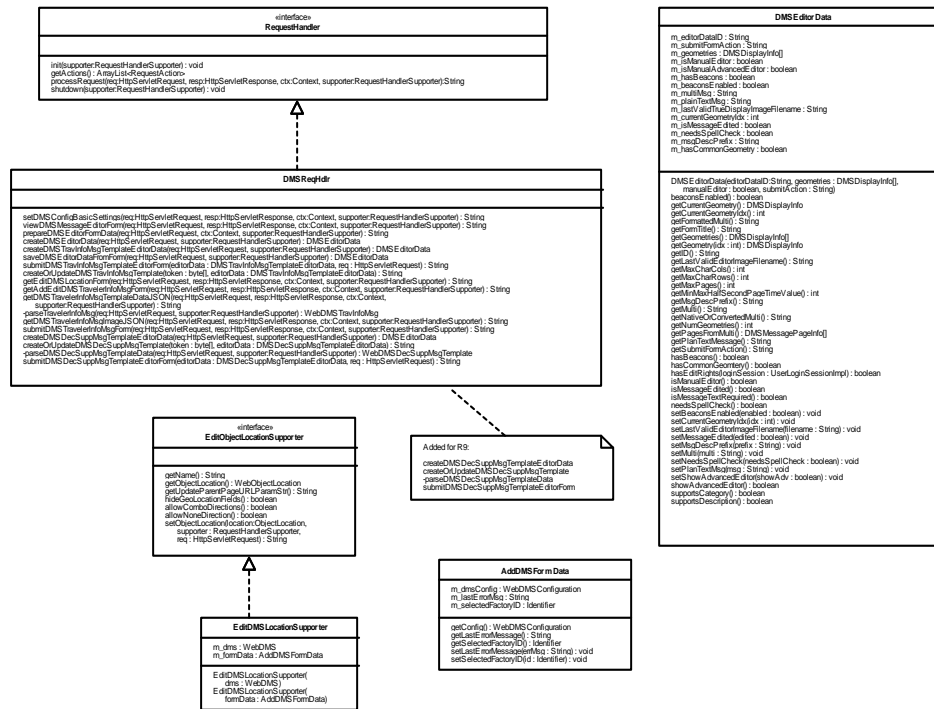
This diagram shows the processing to view the Response Plan Preview Map. If the call to `lookupRequestedEvent` fails to locate the `WebTrafficEvent` from the request, an error is obtained. Otherwise, a call to `getResponsePlanDecSuppData` retrieves the recommended and not recommended DMSs lists and loads the `ResponsePlanPreviewMap` page into the context.



**Figure 5-77. ResponsePlanReqHdlr:viewResponsePlanPreviewMap (Sequence Diagram)**

### 5.12.1.1 GUIDMSServletClasses (Class Diagram)

This diagram shows CHART GUI servlet classes related to dynamic message signs.



**Figure 5-78. GUIDMSServletClasses (Class Diagram)**

#### 5.12.1.1.1 AddDMSFormData (Class)

This class represents the data in the Add DMS and Copy DMS forms.

#### 5.12.1.1.2 DMSEditorData (Class)

This class represents an instance of a DMS message being edited in an editor. It provides storage so that the message and editor state can be preserved during interim requests before the form is submitted. It also has logic for manipulating the editor session. This is a base class and will be extended for specific editor types.

#### 5.12.1.1.3 DMSReqHdlr (Class)

This class is a request handler used to process requests related to dynamic message signs (DMS).

#### **5.12.1.1.4 EditDMSLocationSupporter (Class)**

This class is used to support editing the location of an existing or new DMS.

#### **5.12.1.1.5 EditObjectLocationSupporter (Class)**

This interface provides functionality allowing the location data to be edited. (For example, the target of the edited location may be an existing object, or it may be a form data object for creating a new object).

#### **5.12.1.1.6 RequestHandler (Class)**

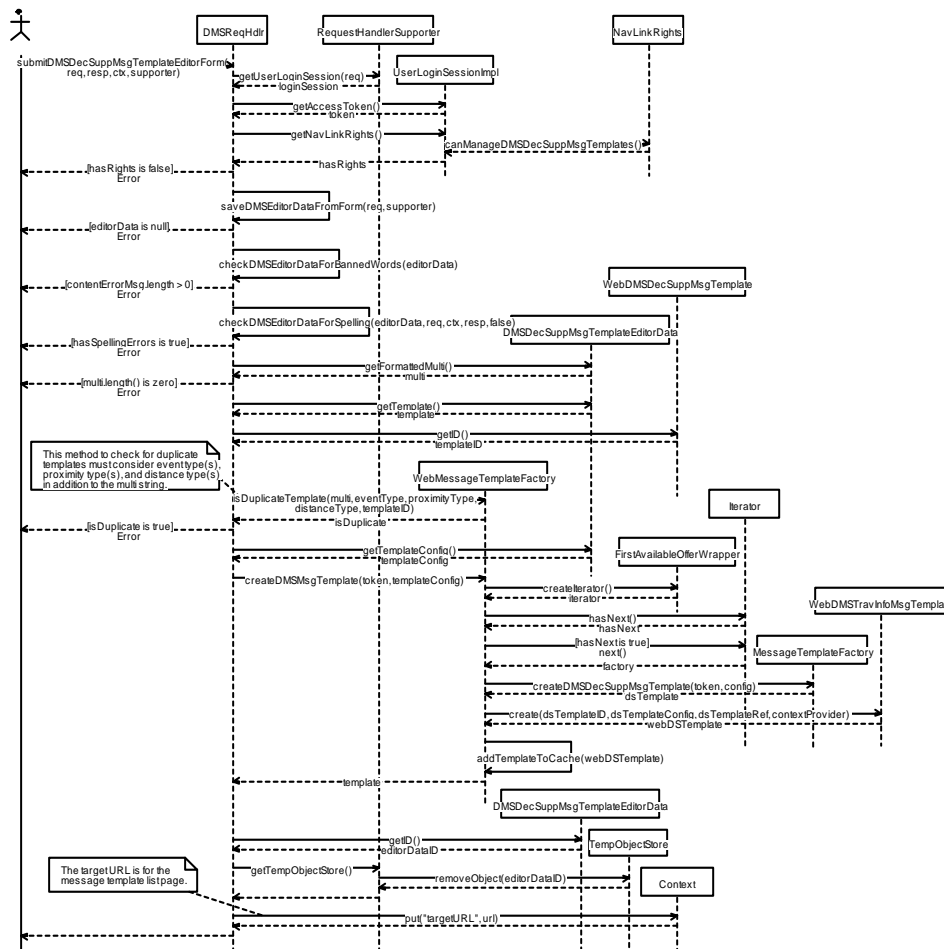
This interface specifies methods that are to be implemented by classes that are used to process requests.



## 5.12.2 Sequence Diagrams

### 5.12.2.1 chartlite.servlet.dms.DMSReqHdlr:addDMSDecSuppMsgTemplate (Sequence Diagram)

This diagram shows the processing that occurs when the editor template form is submitted for adding a DMS decision support message template. The login session is checked to verify the user has the functional rights to add a message template. The editor data is saved to the temporary object store. The message is checked for banned words and spelling errors. The template ID, description, event type(s), proximity type(s), and distance type(s) are used to check for the existence of a duplicate template in the template factory. A new template (based on the configuration from the template editor) is created using the template factory. The new template is added to the template factory cache. The editor data is removed from the temporary object store. A response is sent to redirect the browser to the updated list of templates.



**Figure 5-79. chartlite.servlet.dms.DMSReqHdlr:addDMSDecSuppMsgTemplate (Sequence Diagram)**

### 5.12.2.2 chartlite.servlet.dms.DMSReqHdr:editDMSDecSuppMsgTemplate (Sequence Diagram)

This diagram shows the processing that occurs when the editor template form is submitted for editing a DMS decision support message template. The login session is checked to verify the user has the functional rights to edit a message template. The editor data is saved to the temporary object store. The message is checked for banned words and spelling errors. The template ID, description, event type(s), proximity type(s), and distance type(s) are used to check for the existence of a duplicate template in the template factory. The template configuration is updated based on the configuration from the template editor. The template is updated in the template factory cache. The editor data is removed from the temporary object store. A response is sent to redirect the browser to the updated list of templates.

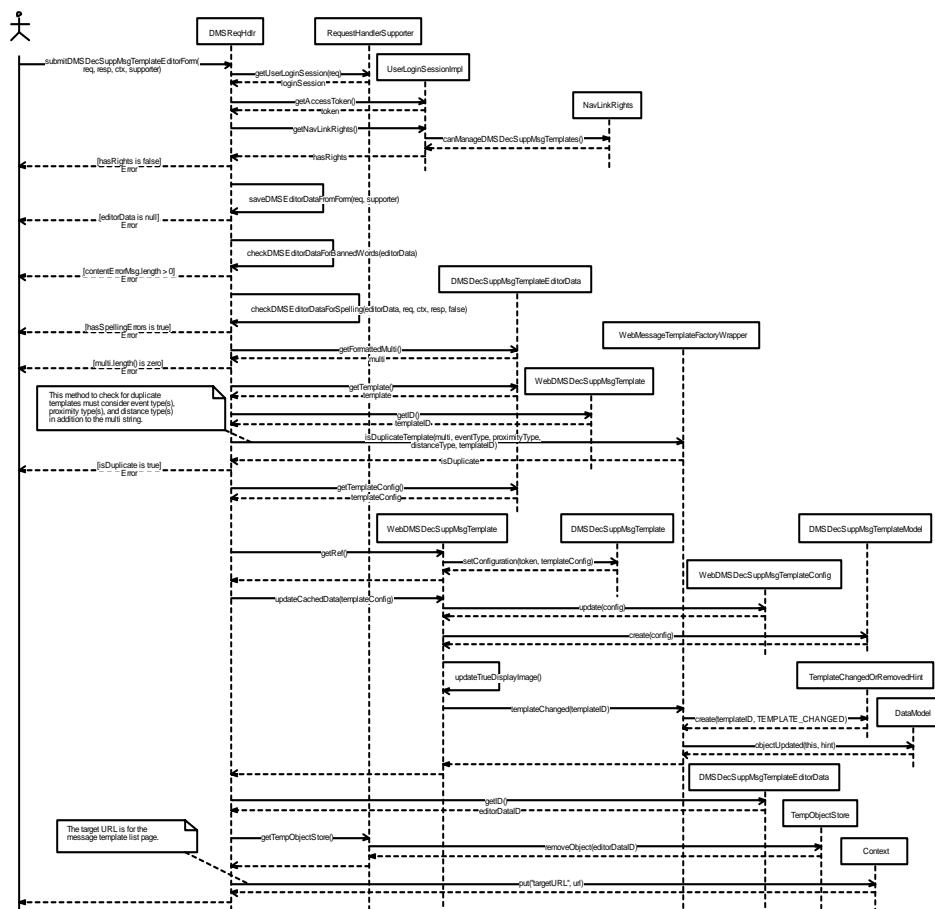


Figure 5-80. chartlite.servlet.dms.DMSReqHdr:editDMSDecSuppMsgTemplate (Sequence Diagram)



process requests.

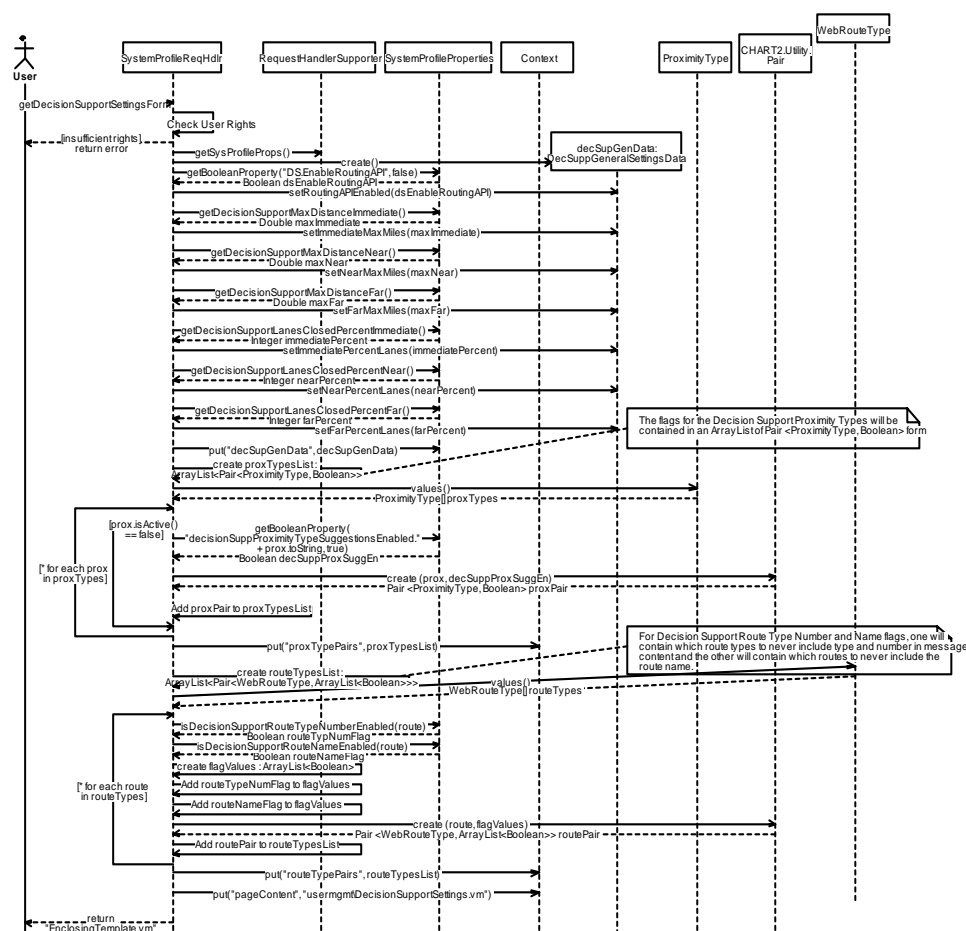
#### **5.13.1.1.4 SystemProfileReqHdlr (Class)**

This class is a request handler that processes requests related to the system profile.

## 5.13.2 Sequence Diagrams

### 5.13.2.1 SystemProfileReqHdr:getDecisionSupportGeneralSettingsForm (Sequence Diagram)

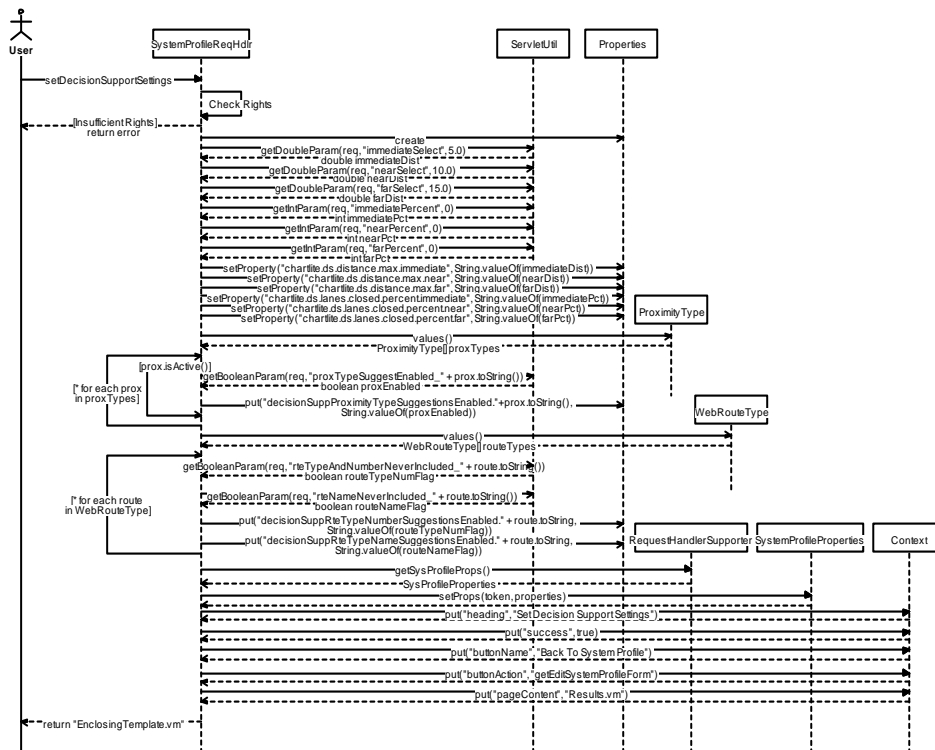
This diagram shows the processing to populate and display the Decision Support General Settings Form Page under the System Profile Settings Page. If they have not been granted the right to change the system profile, an error message is shown. Otherwise, the Max distances (Immediate, Near, Far) for a DMS from the Traffic Event as well as the lane closure percentages for each distance are retrieved from the SystemProfileProperties and populated within the context. The available proximity types and the current set flags are also retrieved from the System Profile and populated in the context within an ArrayList of ProximityType and Boolean pairs. For each of the current route types, the current configured values for route number and route name are retrieved from the System Profile and populated in the context within an ArrayList of Pairs of WebRouteTypes and a 2-Field Array of boolean flags for route type number and route name.



**Figure 5-82. SystemProfileReqHdr:getDecisionSupportGeneralSettingsForm (Sequence Diagram)**

### 5.13.2.2 SystemProfileReqHdr:setDecisionSupportGeneralSettings (Sequence Diagram)

This diagram shows the processing that is performed when the form used to set the Decision Support General Settings is submitted. If the user has not been granted the right to change the system profile, an error message is shown. Otherwise, a Properties object is created and populated with the Maximum Immediate, Near, and Far DMS distances and Lane Closure percentages from the request parameters. The proximity type settings for the DMS to the Traffic Event and the Route Type / Number and Route Name settings for message suggestion are also retrieved from the request parameters and populated within the Properties object. A call to setProps updates the SystemProfileProperties with the Properties object. Navigation back to the System Profile Page is also added for the user.



**Figure 5-83. SystemProfileReqHdr:setDecisionSupportGeneralSettings (Sequence Diagram)**



#### **5.14.1.1.5 CHARTLayers. SpecifyLocationMap (Class)**

This class is used for specifying the location of an object. It supports a feature (and marker) to represent the location specified by the user. When an object location is specified with a proximity of "BETWEEN" or "FROM-TO", the map supports a marker at the beginning and ending features. The map interacts with a Flex application containing a Specify Location form in both directions, to allow the map to update the form and vice versa. This map will support panning and zooming.

#### **5.14.1.1.6 EventLauncherMap (Class)**

This map extends the SpecifyLocationMap to add functionality specific to the Event Launcher.



## 5.14.2 Sequence Diagrams

### 5.14.2.1 ResponsePlanPreviewMap:handleMapDataJSON (Sequence Diagram)

This shows the processing when the Response Plan Map receives the JSON response. For each layer in the JSON object, the type of layer is retrieved and the features are updated. For any layer that is in the map, but is not represented in the JSON object the features are removed. A call to showCHARTObject is also made to show the area around the traffic event.

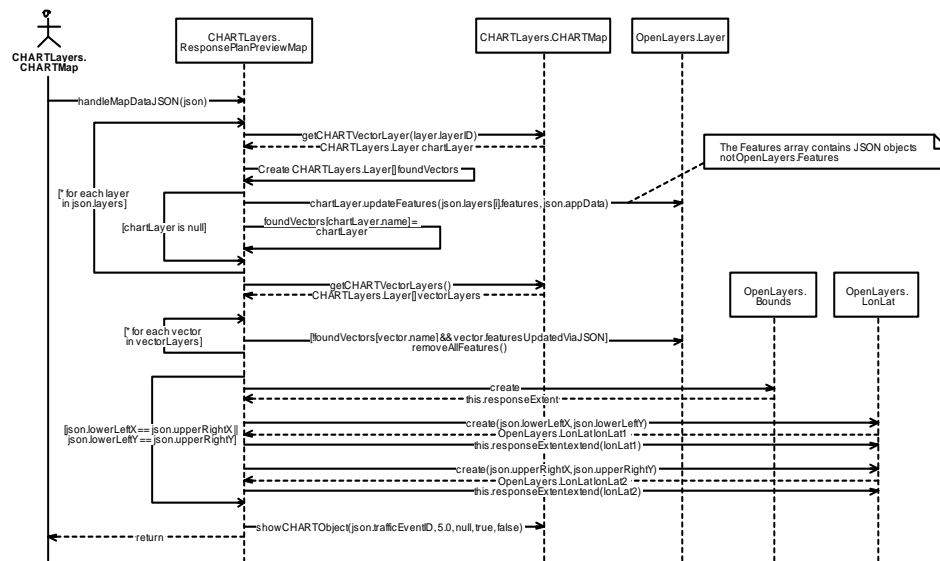


Figure 5-85. ResponsePlanPreviewMap:handleMapDataJSON (Sequence Diagram)

### 5.14.2.2 ResponsePlanPreviewMap:initialize (Sequence Diagram)

This shows how the Response Plan Preview map is initialized. The base class is called first to initialize the map's basic functionality. The base layers are added as well as the vector layers for the DMSs, DMSs (Recommended), DMSs (Not Recommended), and HARs layers. The controls are added, including the layer switcher, tooltips, navigation control, and hover control for popups. Event Handlers are registered for the highlight feature and for the popup click controls.

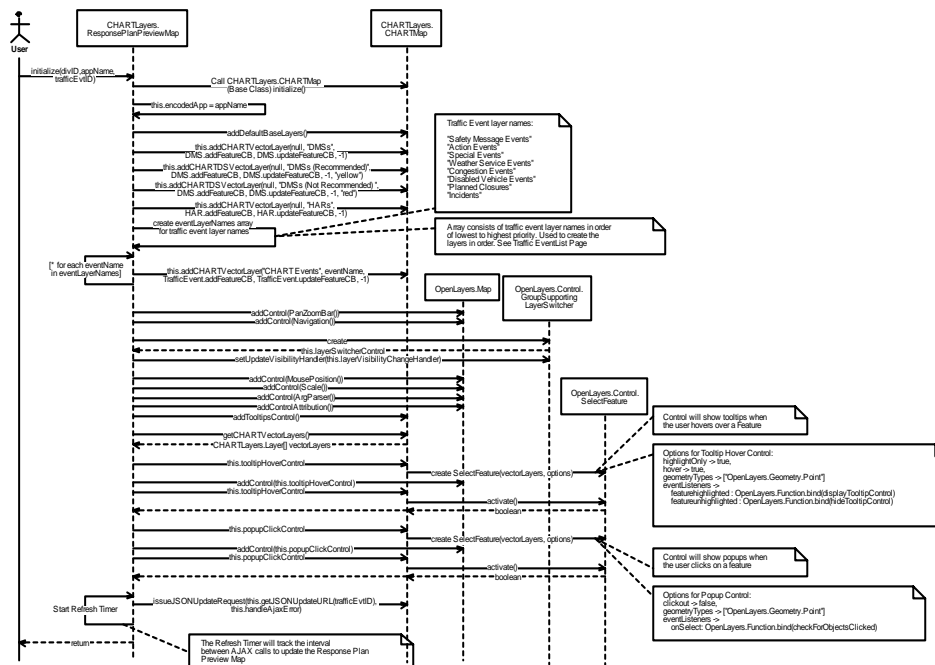


Figure 5-86. ResponsePlanPreviewMap:initialize (Sequence Diagram)

## 6 Use Cases – Desktop Video

The following use case diagrams depict new functionality for the displaying video on the desktop and also identify existing features that will be enhanced. The use case diagrams for this feature exist in the Tau design tool in the Release9 area. The sections below indicate the title of the use case diagrams that apply to this feature.

### 6.1 R9VideoOnDesktopUses (Use Case Diagram)

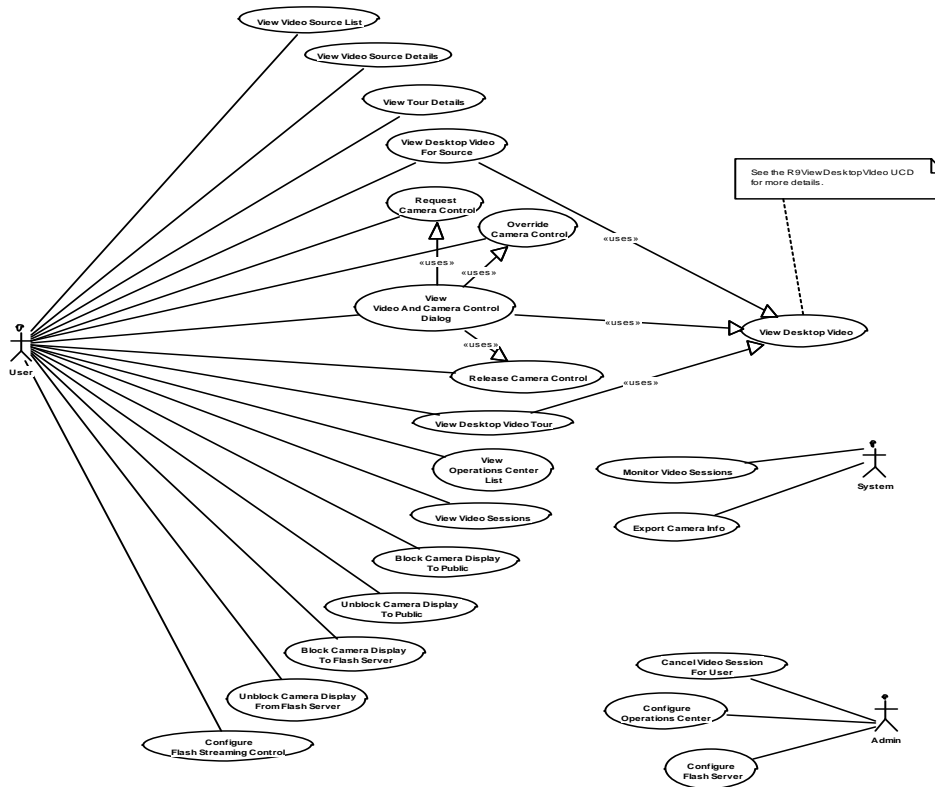


Figure 6-1. R9VideoOnDesktopUses (Use Case Diagram)

#### 6.1.1 Administrator (Actor)

An administrator is a CHART user that has functional rights assigned to allow them to perform administrative tasks, such as system configuration and maintenance.

#### 6.1.2 Block Camera Display To Flash Server (Use Case)

A user with the Block Display To Web And Media right will be able to block a camera from being displayed on a specified SFS server (public or non-public) for which the camera is configured.

### **6.1.3 Block Camera Display To Public (Use Case)**

A user with the Block Display To Web And Media right can block a camera from being displayed on public monitors and on streaming flash servers (SFS) associated with the camera that are designated as "public". This will stop any current streaming of the camera, and will prevent future streaming as long as the camera is blocked. (Prior to R9, this affected ALL SFSs configured for a camera (although only the public ones were configured), but in R9 a "public" flag will be added so that non-public SFSs can be configured for a camera and these will be skipped when a camera is blocked to public.)

### **6.1.4 Cancel Video Session For User (Use Case)**

A user with the Cancel Video Session right will be able to cancel the video session for another user, to release the video session resource so that others can use it. The user's browser will be informed that the session has been cancelled, at which time it will stop the video, display an error message, and close the video window.

### **6.1.5 Configure Flash Server (Use Case)**

A user with the Configure System right will be able to add or modify a Streaming Flash Server (SFS) configuration. The user can specify the server name, IP, port, username, password, and (new for R9) a "public" flag that indicates whether the SFS server will be automatically blocked if the Block To Public command is issued for a camera.

### **6.1.6 Configure Flash Streaming Control (Use Case)**

The system shall allow an administrator to specify the settings for controlling a Flash Streaming Server for a video source. The user can select from the SFS configurations defined for the system to populate this information. The user will be able to specify the Streaming Flash Server IP, control port, password, and (new for R9) a flag indicating whether the SFS is to be considered "public" for the purposes of blocking.

### **6.1.7 Configure Operations Center (Use Case)**

A user with the Configure System right can configure (add or modify) an operations center. For R9, a setting will be added to limit the total number of simultaneous video sessions that all users logged into that operations center are allowed to have open.

### **6.1.8 Export Camera Info (Use Case)**

The system will export camera information for use by external systems, specifically CHARTWeb and Intranet Map. For R9 this will include name (e.g. Internal, Mobile, SWGI, Public), host/ip, port, and an indication if it provides video streams to the public. CHART tour configurations are not exported.

### **6.1.9 Monitor Video Sessions (Use Case)**

The system will monitor the desktop video sessions. This includes enforcing the per op center limit, as well as querying and updating the status of active sessions as reported by all GUIs, and cleaning up abandoned sessions that are no longer in use.

#### **6.1.10 Override Camera Control (Use Case)**

An operator with the Override Camera Control right may override control of a camera that is currently being controlled by another user, which terminates the original user's control session. In R9, this will no longer require the camera to be displayed on a local monitor before overriding control. As of R9, control will be allowed if the camera is displayed in one of the user's desktop video sessions. If the user overrides control and the camera is not displayed on a local monitor or in one of the user's existing video sessions, a video session will automatically be opened (if the user has the View Desktop Video right and the camera is eligible for display). If the attempt was successful to acquire the video session, the video will start playing automatically on the video / control dialog; otherwise, an error will be displayed to the user and control will not be granted.

#### **6.1.11 Release Camera Control (Use Case)**

A user with a video / control dialog that is showing both desktop video and the camera control form will be able to release the camera control session without ending the video session. The system will also release the camera control session when the dialog is closed, if it was not previously released.

#### **6.1.12 Request Camera Control (Use Case)**

An operator with the Request Camera Control right may request control of a camera. This means that the operator may send pan/tilt/zoom (PTZ) and other commands to the camera. In R9, this will no longer require the camera to be displayed on a local monitor before requesting the session. As of R9, control will be allowed if the camera is displayed in one of the user's desktop video sessions. If the user requests control and the camera is not displayed on a local monitor or in one of the user's existing video sessions, a video session will automatically be opened (if the user has the View Desktop Video right and the camera is eligible for display). If the attempt was successful to acquire the video session, the video will start playing automatically on the video / control dialog; otherwise, an error will be displayed to the user and control will not be granted.

#### **6.1.13 System (Actor)**

The System actor represents any software component of the CHART system. It is used to model uses of the system which are either initiated by the system on an interval basis, or are an indirect by-product of another use case that another actor has initiated.

#### **6.1.14 Unblock Camera Display To Public (Use Case)**

A user with the Block Display To Web And Media right can unblock a camera that was previously blocked from being displayed on public monitors and on streaming flash servers (SFS) designated as "public". (Prior to R9, this affected ALL SFSs configured for a camera (although only the public ones were configured), but in R9 a "public" flag will be added so that non-public SFSs can be configured for a camera and these will be skipped when a camera is unblocked to public.)

#### **6.1.15 Unblock Camera Display From Flash Server (Use Case)**

A user with the Block Display To Web And Media right will be able to unblock a camera that was previously blocked from being displayed on a specified SFS server (public or non-public) for which the camera is configured.

#### **6.1.16 User (Actor)**

The User class represents a Chart II system user. In order to log into the Chart II system, a user must be defined in the user database.

#### **6.1.17 View Operations Center List (Use Case)**

The user will be able to view a list of operations centers. For R9, this list will be enhanced to include the total number of video sessions that are allowed to be used by users logging into the center, as well as the number of active video sessions that are actually in use.

#### **6.1.18 View Video And Camera Control Dialog (Use Case)**

A user with the Request or Override Camera Control right and optionally the View Desktop Video right will be able to view a video and/or control window for a controllable camera with a Streaming Flash Server configured for the camera that matches the GUI's designated SFS server and the camera is otherwise eligible for display. The user will be able to invoke it either in video mode or control mode, assuming the user has the appropriate right. If invoked in video mode, the video will automatically be played. If invoked in control mode, the video will also be displayed automatically if the camera is not already displayed on a local monitor or in another of the user's desktop video sessions (and if the user has the View Desktop Video right and the camera is eligible for desktop video display). The user will be able to switch modes in the dialog to display the desktop video session or control form (if not already displayed). If both the video session and control form are displayed, the user will be able to release control or release the video session (but the user will only be allowed to close / end the video session while retaining control if the camera is displayed on a local monitor or in another of the user's desktop video sessions). The camera name and location will also be displayed in this dialog.

#### **6.1.19 View Desktop Video (Use Case)**

A user with the View Desktop Video right will be able to view desktop video. The system will acquire a "Video session" resource from the operations center before the video is played, monitor the eligibility status of the video sources and session throughout the session, and end / release the video session when the window is closed (or the session is canceled by another user). The video for a video source (either standalone or within a tour) will be displayed only if it is online, is not blocked to the SFS server assigned to the user's GUI, and is not blocked to the organization assigned to the user's operations center. While the video window is open, the user can play, pause, or resize the video. (For more details see the detailed use cases). The user will be allowed to open multiple video windows on the same computer (as long as it does not exceed the limit of the user's operations center).

### **6.1.20 View Desktop Video For Source (Use Case)**

A user with the View Desktop Video right will be able to view the video for a single video source that is not a controllable camera (or is a controllable camera that the user doesn't have rights to control) if a Streaming Flash Server configured for the video source that matches the GUI's designated SFS server and the source is otherwise eligible for display. The name and location of the video source will be shown, in addition to the video image and controls. The video will be automatically played when it is displayed.

### **6.1.21 View Desktop Video Tour (Use Case)**

A user with the View Desktop Video right will be able to view a desktop video tour (which is a sequence of streaming camera images), if a video session is available from the user's operations center's pool of sessions. The user will be able to select a preconfigured tour for display on the desktop. Only video sources with streams configured for them will be included in the desktop tour, and only those that are online and not blocked (either to the public or to the user's operations center's organization) are eligible for display. If the tour has presets, the system will display a message to the user indicating that the presets will be ignored. If a preconfigured tour is displayed, the name of the tour will also be displayed in addition to the video.

### **6.1.22 View Tour Details (Use Case)**

A user with sufficient rights may view the configuration and status details for a preconfigured video tour. For R9 this existing functionality will be enhanced to show the video sessions that are displaying the tour (in addition to the monitors displaying the tour, which were shown in previous releases). For each video session the user name, operations center, and browser host / IP will be displayed. If the tour has presets, the system will indicate that the presets will be ignored if the tour is played via desktop video.

### **6.1.23 View Video Sessions (Use Case)**

A user will be able to view a list of the video sessions that exist in the system. For each video session, the following information will be displayed: name of the video source or tour, hostname or IP of the desktop displaying the session, the hostname or IP of the session provider, the name of the user, the name of the operations center that the user is logged into, and the date/time when the session was started, and the date/time when the session was last confirmed to be in use. The user will be able to sort and/or filter the list by any of these values. By default, the list will be filtered to display only the sessions owned by the user viewing the list.

### **6.1.24 View Video Source Details (Use Case)**

A user may view the configuration and status details for a video source such as a camera. For R9 this existing functionality will be enhanced to show the desktop video sessions that are displaying the source (in addition to the monitors displaying the source, which were shown in previous releases). For each video session the user name, operations center, and browser host / IP will be displayed. The "Display Blocked To Public" status that existed prior to R9 will be modified to indicate "Display Blocked To Monitors" and a separate

status will be listed for each Streaming Flash Server which includes the SFS name (or IP, if name not available), a flag indicating whether the SFS is public or not, and an indicator stating whether or not the source is blocked in that SFS.

#### **6.1.25 View Video Source List (Use Case)**

A user may view the list of video sources (cameras, tours, and other sources). The existing functionality will be enhanced for R9 so that the "Local Displays" column will also indicate when the user has a video session for the video source or tour.



## 6.2 R9ViewDesktopVideo (Use Case Diagram)

This diagram shows the detailed use cases that are involved when a user views a desktop video session.

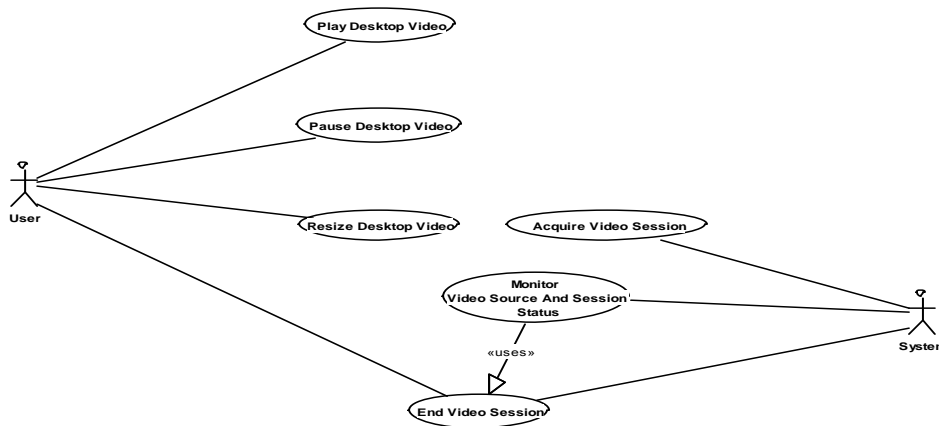


Figure 6-2. R9ViewDesktopVideo (Use Case Diagram)

### 6.2.1 Acquire Video Session (Use Case)

The system will acquire a video session resource from the user's operations center before showing a video stream on the desktop. This will decrease the number of video sessions available to other users by one. If no video sessions are available, an error message will be displayed to the user and the video will not be played.

### 6.2.2 End Video Session (Use Case)

The system shall end (or release) the video session resource after the user closes the window (or portion of the window) that is displaying the corresponding video. The system also will end a video session if the session was canceled by another user. The system will also automatically end a video session if the session is somehow abandoned if a window is closed unexpectedly. When a video session is ended, the session will be released from the user's operations center's limited number of video sessions, thereby making a video session available for someone else at the operations center to use.

### 6.2.3 Monitor Video Source And Session Status (Use Case)

The system will monitor the eligibility status of video sources being played in a video window, as well as the eligibility of the video session. A video source is considered eligible if a video stream is configured for on the GUI's designated SFS, the camera is online or in maintenance mode, and the video stream is not blocked to the GUI's designated SFS, and is not blocked to the user's op center's organization. If displaying a single video source and that source is no longer eligible, the video will be stopped and the system will

indicate to the user that the video source is not available. If displaying a tour and a source is no longer eligible, that source will no longer be displayed as long as it remains ineligible, and it will be skipped when switching sources. If an ineligible source becomes eligible again while the video session is active, that source will be displayed again. If playing a tour and all sources are ineligible, the system will indicate to the user that no sources are available. If the user's video session was canceled by another user, the video will stop playing, an error message will be displayed and the video session will end.

#### **6.2.4                      Pause Desktop Video (Use Case)**

The user will be able to pause the video if it is playing. When paused, the image at the time of pausing will remain displayed. (If pausing a tour, that video source will remain selected during and after the pause, until the next switch, unless it becomes ineligible). Usually the video session will NOT be released while the video is paused (unless the session is canceled by another user), which allows for play to resume without requesting a video session resource again.

#### **6.2.5                      Play Desktop Video (Use Case)**

The user will be able to play video from within a video session if the video was previously paused or not already playing. The system will also play video automatically when first displaying a requested video screen. If playing video after previously pausing it, the video will resume at as close to the current time as possible (i.e., the video will behave as a Stop / Start rather than continuing to play from the paused time). If playing video for a video tour after previously pausing the tour, the playback will resume on the same video source that was displayed when the user paused playback. When playing a tour, the system will play only those sources that are eligible for display (i.e., those that are not offline, with streams that are not blocked to the GUI's designated SFS server or blocked to the user's operations center's organization) and will ignore any others. Any presets defined in a pre-configured tour will be ignored (to avoid possible contention in the camera communications).

#### **6.2.6                      Resize Desktop Video (Use Case)**

A user will be able to resize the video that is playing on their desktop. This may include a choice of several possible sizes. The user will also be able to enter full screen mode, and the video will expand to fill the entire screen (but only one monitor if a multi-monitor display). The user will also be able to exit full screen mode, at which time the video will return to the size that was being used prior to entering full screen mode.

#### **6.2.7                      System (Actor)**

The System actor represents any software component of the CHART system. It is used to model uses of the system which are either initiated by the system on an interval basis, or are an indirect by-product of another use case that another actor has initiated.

#### **6.2.8                      User (Actor)**

The User class represents a Chart II system user. In order to log into the Chart II system, a user must be defined in the user database.

## 7 Detailed Design – Desktop Video

### 7.1 Human-Machine Interface

This section describes screens that are new or changed for CHART Release 9 to support displaying video sources and tours on the user's desktop.

#### 7.1.1 Video Sources

The Video Source List page will have a few minor changes. First, the Cameras list:



All Video Sources			
<a href="#">Show Cameras Only</a> <a href="#">Cameras</a> <a href="#">Video Tours</a> <a href="#">Other Video Sources</a>			
Cameras (6) <a href="#">Add Camera</a> <a href="#">Set Columns</a> <a href="#">(show default columns)</a>			
Description <sup>Δ</sup> / Location	Actions	Region --Any--	Local Displays
 <a href="#">SIM COHU 3955 2</a> (Location Unspecified)	<a href="#">Request Control</a> <a href="#">Display on Desktop</a> <a href="#">Display on Monitors</a>		<a href="#">SIM mon 1</a> <a href="#">SIM mon 12</a>
 <a href="#">SIM COHU A</a> I-270 N OF DEMOCRACY BLVD	<a href="#">Request Control</a> <a href="#">Display on Desktop</a> <a href="#">Display on Monitors</a>		<a href="#">SIM mon 2</a> <a href="#">(DESKTOP)</a>

Figure 7-1 Video Sources: Cameras

The changes for the Cameras list include:

- 1.The **Display** link is renamed to **Display on Monitors**.
- 2.A new **Display on Desktop** link is added. Clicking on this will bring up a video window, which can be used as video only or the user can request control of the camera from that window.
- 3.In the **Local Displays** column, an indicator shows that the camera is displayed on the desktop.
- 4.The **Request Control** link will no longer require the camera to be displayed on a local monitor. If the user requests control of the camera and the camera is not displayed on a local monitor, a video session will automatically be opened on the Camera Control Dialog.

The Video Tours and Other Video Sources portion of the page will also have similar changes:

Video Tours (1) [Configure](#)

Click on a Category Name to expand or contract the list of tours

[newcat](#) (1 tour)

Name	Actions	Displayed On
<a href="#">SIM cameras</a>	<a href="#">Display on Monitors</a> <a href="#">Display on Desktop</a>	(DESKTOP)

[Cameras](#)[Video Tours](#)[Other Video Sources](#)

Other Video Sources (2) [Add Video Source](#)

Description	Local Displays	Status	Owning Org
 <a href="#">IP NVA</a>	<a href="#">SIM mon 5</a> (DESKTOP)	Online	AOC
 <a href="#">NVA</a>		Online	AOC

**Figure 7-2 Video Sources: Tours and Other Sources**

The changes for the Tours list are:

- 1.The **Display** link is renamed to **Display on Monitors**.
- 2.A new **Display on Desktop** link is added. Clicking on this will bring up a video window.
- 3.In the **Displayed On** column, an indicator will show if the tour is displayed on the desktop.

The change for the Other Video Sources list is that in the **Local Displays** column an indicator will show if the video source is displayed on the desktop.

## 7.1.2 Camera / Video Source Details

### COHU 3955 Camera : SIM COHU A


I-270 N OF DEMOCRACY BLVD

#### Status

Controlled By:

Operational Status:

Displayed On:

Monitor	Group	Owned By	Action
 <a href="#">SIM mon 2</a>	<a href="#">group 2</a>	AOC	<a href="#">Remove Image</a>

#### Actions

[Take Offline](#)

[Request Control](#)

[Display on Desktop](#)

[Display on Monitors](#)

[Poll Now](#)

[Unblock Display To Public](#)

[Set Display Status to Failed](#)

[Set Control Status to OK](#)

[Edit Blocked Display Organizations](#)

[Edit Blocked Control Organizations](#)

[Copy](#)

#### Video Sessions:

User	Op Center	Browser Host/IP	Last Used	Actions
admin	SOC	192.168.1.5	18:18	<a href="#">Remove</a>

Directly Connected To: None

Last Status Time: Device Status: N/A  
Monitor Status: 06/16/11 12:04

Display Blocked To Monitors: YES

Flash Server Status For This Camera:

Name	Public?	Status	Action
Internal	NO	Not Blocked	<a href="#">Block</a>
SWG1	YES	BLOCKED	<a href="#">Unblock</a>
Public	YES	BLOCKED	<a href="#">Unblock</a>
Mobile	YES	BLOCKED	<a href="#">Unblock</a>

Figure 7-3 Video Source Details

The Video Source Details page has several changes:

1. A new **Video Sessions** table will be added that lists the video windows of all users currently viewing the video source. This displays information to identify the user, his/her location, and the last time the video window was known to be open.
2. A user with the right to end other users' video sessions can click on the **Remove** link to remove (end) the video session, which may be useful to free up the video session resource so that other users can use it. This will cause the video to be stopped on the user's desktop within a short time.
3. The **Display Blocked** status will be changed to **Display Blocked To Monitors**, which indicates whether the display has been blocked / unblocked to monitors. (In previous releases, this also represented the blocking status of the video source's stream on all SFS servers, but that status is now being split into a separate table because in R9 not all SFS servers will be necessarily blocked).
4. A new **Flash Server Status For This Camera** table lists all configured SFS servers, whether they are classified as "public" or not, and the blocked / unblocked status for the camera's stream on that server. (Note however, that the SFS server currently does not support querying the stream status, so the status displayed here will not reflect any block / unblock commands made from outside of the CHART application (such as from the Skyline application). It is assumed that any block / unblock commands will be issued via CHART only).
5. In previous releases only public SFS servers were configured within CHART, so that the **Block to Public** command affected all SFS servers that were configured for the video source. In R9, non-public SFS servers may also be configured and these will not be affected by the Block to Public command.
6. A user with appropriate rights will be able to **Block** or **Unblock** the video source's stream on a single SFS server by clicking on the corresponding link in the table.
7. The same changes will be made to the Actions links on the Video Source Details page that affect the Video Source List page:
  - a. The **Display** link is renamed to **Display on Monitors**.
  - b. A new **Display on Desktop** link is added. Clicking on this will bring up a video window, which can be used as video only, or (if it is a controllable camera) the user can request control of the camera from that window.
  - c. The **Request Control** link will no longer require the camera to be displayed on a local monitor. If the user requests control of the camera and the camera is not displayed on a local monitor, a video session will automatically be opened on the Camera Control Dialog.

### 7.1.3 Tour Details

### Video Tour: SIM cameras

#### Status

Displayed On Monitors:

Monitor Name	Action
Tour Is Not Displayed on Any Monitors	
Display on Additional Monitors	
<a href="#">Select</a>	<input type="text"/> <a href="#">Search</a>

Video Sessions:

User	Op Center	Browser Host/IP	Last Used	Actions
admin	SOC	192.168.1.5	11:55	<a href="#">Remove</a>

[Display on Desktop](#)  
Only unblocked, transcoded, online cameras will show when this tour is run on the desktop.

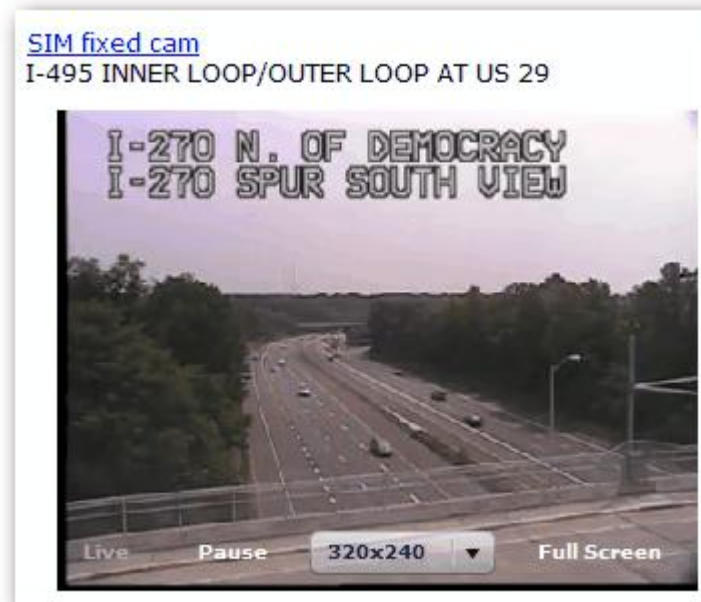
Figure 7-4 Tour Details

The Video Source Details page will have the following changes:

- 1.A **Display on Desktop** link will be added to allow the user to view the tour in a window on the desktop. Note that, as stated in the disclaimer, only eligible cameras that will be displayed in the tour. Also, any presets will be ignored when a desktop tour is run (to prevent an unknown number of running tours from creating communications problems for the cameras).
- 2.A new **Video Sessions** table will be added that lists the video windows of all users currently viewing the tour. This displays information to identify the user, his/her location, and the last time the video window was known to be open.

#### 7.1.4 Video Popup (new)

When the user displays video for a tour or a video source / camera that is not controllable, a simple video popup window will be displayed. This window is new for R9. The two variants are similar in appearance, differing only in the title.



**Figure 7-5 Video Popup (Video Source / Non-Controllable Camera)**

The Video Popup for a plain video source (or camera that can't be controlled) consists mainly of a video component, which also serves as common component in the the video windows described below. This video component has the following controls:

- The **Pause** button pauses the video.
- The **Live** button unpauses the video if it is paused, causing live video to be streamed.
- The video size selection box allows the user to select from various sizes for the video.  
(Note that selecting a larger size only increases the resolution up to the native resolution of the video - a larger size than that will not increase the resolution, but a smaller size than the native resolution will cause a loss of resolution).
- The **Full Screen** button causes the window to enter full screen mode, which occupies the whole monitor (but only the current monitor if it is a multi-monitor desktop). When in full screen mode, the user can exit via the Esc key or by clicking on an **Exit Full Screen** button that will appear next to the **Pause** button.



The window for a controllable camera will contain a **Request Control** (or **Override Control**) link if the user has the appropriate functional right. Clicking on this link will cause the window to transform into a combined Video / Control window (see following section).



**Figure 7-6 Video and Control Window (Video Only Mode)**

If displaying a tour, the video will switch to the different cameras / video sources in the tour if they have streams available and are online and unblocked. As noted above, any camera presets configured for the tour will be ignored.



**Figure 7-7 Video Popup (Video Tour)**

### 7.1.5 Camera Control

The Camera Control dialog that existed in previous releases is being changed in R9 to add video support. The dialog will be able to transform between any of three different modes:

- Video Only
- Control Only
- Video and Control

The first of these, Video Only mode, is shown above in the previous section. The window appears in this mode if the user clicks on the **Display on Desktop** link from places such as the Camera List or Camera Details page. However, it can also be shown if the window was previously in Video and Control mode and the user clicked on **Release Control**. There is a **Request Control** or **Override Control** link, which will show the camera control GUI and will cause the dialog to transform into Video and Control mode.

The window can also be shown in Control Only mode (see below) if the user clicks on **Request Control** or **Override Control** from places such as the Camera List or Camera Details page and the camera is displayed on a local monitor. However, it can also be shown if the window was previously in Video and Control mode and the user clicked on **Hide Video**. This version of the window is similar in functionality to pre-R9 version of the Camera Control dialog, but the layout has changed in R9 to make room for the video component to be displayed. For R9, there is a **Display on Desktop** link, which will show video and will cause the dialog to transform into Video and Control mode.

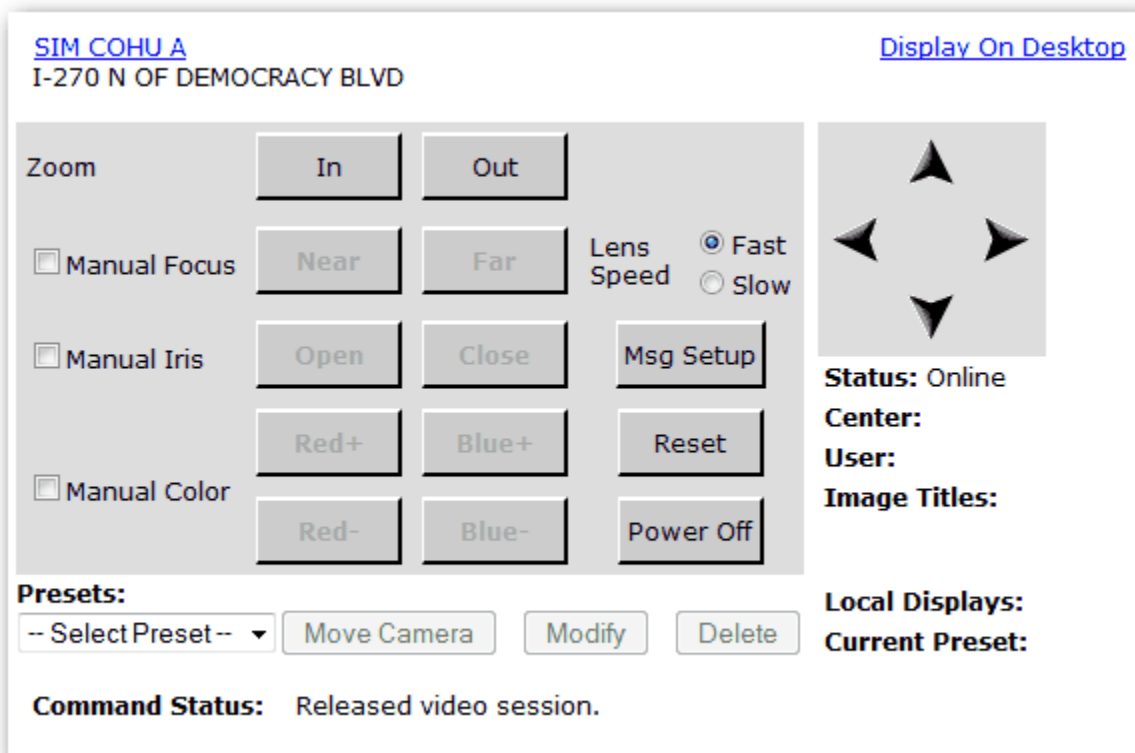


Figure 7-8 Video and Control Window (Control Only Mode)

The third variation of this window, in Video and Control mode, is shown below. This window will appear if the user clicks on **Request Control** or **Override Control** from places such as the Camera List or Camera Details page, and the camera is not already displayed on a local monitor. In that case the video is automatically displayed so that the user can see the results of the camera control operations.

Another way it can enter Video and Control mode is if the window was previously displayed in Video Only mode and the user clicked **Request Control** or **Override Control**, or if the window was previously in Control Only mode and the user clicked on **Display on Desktop**.



**Figure 7-9 Video and Control Window (Video and Control Mode)**

Note that the layout of this window (particularly the button layout) is expected to change during implementation, as the layout as shown above takes up too much screen real estate on the minimum target screen resolution (1024 x 768).

### 7.1.6 Camera Map Popup

A **Display on Desktop** link will be added to the map popup for a camera, so that desktop video can be displayed from the map. This will be available from both the Home Page Map, and the Close Devices Map for a traffic event.

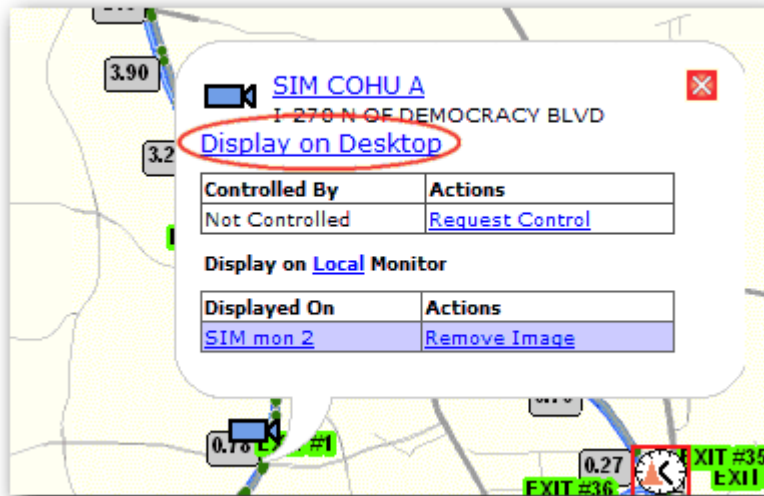


Figure 7-10 Camera Map Popup

### 7.1.7 Operations Centers

The Operations Centers list will have two new columns added for R9: Active Video Sessions and Max Video Sessions. To limit bandwidth usage, an operations center will have a configurable maximum number of video sessions that users logged into that center can have open. The Active Video Sessions column indicates how many sessions are in use.




Operations Centers (3) <a href="#">(Add)</a>				
Name	Traffic Event Summary	Users Logged In	Active Video Sessions	Max Video Sessions
 <a href="#">RITIS</a>	1 Disabled Vehicle, 1 Incident, 1 Congestion Event, 1 Planned Roadway Closure, 1 Action Event, 1 Special Event, 1 Weather Service Event <a href="#">(View)</a>			20
 <a href="#">SOC</a>	7 Disabled Vehicles, 19 Incidents, 2 Congestion Events, 4 Planned Roadway Closures, 2 Safety Events, 4 Special Events, 7 Weather Service Events <a href="#">(View)</a>	1: ( admin )	2	20
 <a href="#">IOC3</a>				20

Figure 7-11 Operations Centers



### 7.1.8 Operations Center Details

The Operations Center Details page will show the setting for maximum active video sessions. (Note that the layout of the page is also reorganized to be more consistent with other details pages in the application).

### Operations Center: SOC

[Remove Operations Center](#)

**Basic Settings:** [\(Edit\)](#)

<b>Name:</b>	SOC
<b>Owning Organization:</b>	SHA
<b>Network Connection Site:</b>	jdp4
<b>Default Local Monitor Group:</b>	NONE
<b>Max Active Video Sessions:</b>	20

**Backup Centers:** [\(Edit\)](#)

| No Backup Op Centers Configured

**Eligible Participants:**

(The participants available to an operator when adding a participant to a traffic event include the eligible participants from the user's operations center and the eligible participants from the event's controlling operations center.)

Name	Type	Action
Animal Control	Agency	<a href="#">Remove</a>
another	Agency	<a href="#">Remove</a>
CHART	Agency	<a href="#">Remove</a>
Fireboard	Agency	<a href="#">Remove</a>
Local - Maint.	Agency	<a href="#">Remove</a>
Local Police	Agency	<a href="#">Remove</a>
MDE	Agency	<a href="#">Remove</a>
MDTA - Maint.	Agency	<a href="#">Remove</a>

**Agency:**

**CHART Unit:**

**Resource:**

**Special Needs:**

To configure the contents of the selection lists above, click [here](#).

Figure 7-12 Operations Center Details

### 7.1.9 Video Administration

A **Video Sessions** link is being added to the Video Administration page. Clicking on this link will bring up the Video Sessions list (see below).



**Figure 7-13 Video Administration**

### 7.1.10 Video Sessions (new)

The Video Sessions list is being added for R9. The page shows information about all known video sessions that are currently open in the system:

Video Sessions							
<u>Subject</u>	<u>User</u>	<u>Op Center</u>	<u>Browser Host/IP</u>	<u>GUI Host</u>	<u>Started</u>	<u>Last Used</u>	<u>Actions</u>
--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾	--Any-- ▾	
SIM COHU C	jparsons	SOC	192.168.1.5	jdp4	14:27	14:27	<a href="#">Remove</a>
SIM COHU 3955 2	jparsons	SOC	192.168.1.5	jdp4	14:27	14:27	<a href="#">Remove</a>

**Figure 7-14 Video Sessions**

The following information is displayed:

- Subject - The name of the video source or tour that is being displayed.
- User – The user displaying the video.
- Op Center – The operations center at which the user playing the video is logged in.
- Browser Host / IP – The hostname and/or IP address of the user’s computer.
- GUI Host – The hostname of the computer running the GUI that is hosting the video session.
- Started – The date / time when the video session was started.
- Last Used – The date / time when the video session was last known to be displayed. This will be updated periodically as long as the video window is open, so it should be close to the current time unless a video window was somehow closed without ending the session. (This could happen if the user’s browser crashed, etc.) The system will clean up any orphaned sessions after some time, but the timestamp for such a session will remain the same until it is cleaned up by the system.

The user will be able to sort by clicking on the column header link or filter by selecting a value from the dropdown list in the column header, for any of the columns, as shown in the screenshot.

A user with sufficient rights will be presented with the **Remove** link. Clicking on this link will end the user’s video session, causing the user’s video to stop a short time later. The user whose video session was ended will see a message saying that the video session was ended and the name of the user who ended it.

### 7.1.11 Streaming Flash Server Configuration

The Streaming Flash Server Configuration page allows the administrator to configure information that allows CHART to send block / unblock commands to the Streaming Flash Server (SFS).

For R9, the Public attribute will be added to allow CHART to differentiate between public and non-public SFSs when a **Block Display to Public** or **Unblock Display to Public** command is issued for a given video source. Prior to R9, CHART assumed that all SFSs in the system were public, so only public SFSs were actually configured on this page. In R9, non-public SFSs will also be configured on this page as well.

Each video source (e.g., camera) will be configured to use the SFSs defined here, but only if a stream has first been configured on an SFS for the video source via the Skyline SFS software (and transcoding server). Each GUI instance will use a single SFS for ALL of its video streaming (which will be configured via the GUI properties file). The GUI will determine whether its SFS is supported for a given video source by checking the video source's configuration. If the video source is configured to support the SFS used by the GUI, the video source will support streaming video and the host/IP listed on this configuration screen will be used to build the URL for streaming video.

Name	Host/IP Address	Port	Password	Public	
Internal	127.0.0.1	9999	password	<input type="checkbox"/>	<a href="#">Remove</a>
SWGI	127.0.0.1	9999	password	<input checked="" type="checkbox"/>	<a href="#">Remove</a>
Public	127.0.0.1	9999	password	<input checked="" type="checkbox"/>	<a href="#">Remove</a>
Mobile	127.0.0.1	9999	password	<input checked="" type="checkbox"/>	<a href="#">Remove</a>

[Add](#)

[Save Changes](#) [Cancel](#)

Figure 7-15 SFS Configuration

### 7.1.12 Add Operations Center

The Add Operations Center page will be changed in R9 to add the Max Active Video Sessions setting. This is the total number of video sessions that CHART will allow to be open for all users logged into that operations center combined. This setting is intended to help limit the network bandwidth used for video streaming.

**Add Operations Center**

**Name:**

**Owning Organization:**

**Default Local Monitor Group:**

**Max Active Video Sessions:**

**Site**

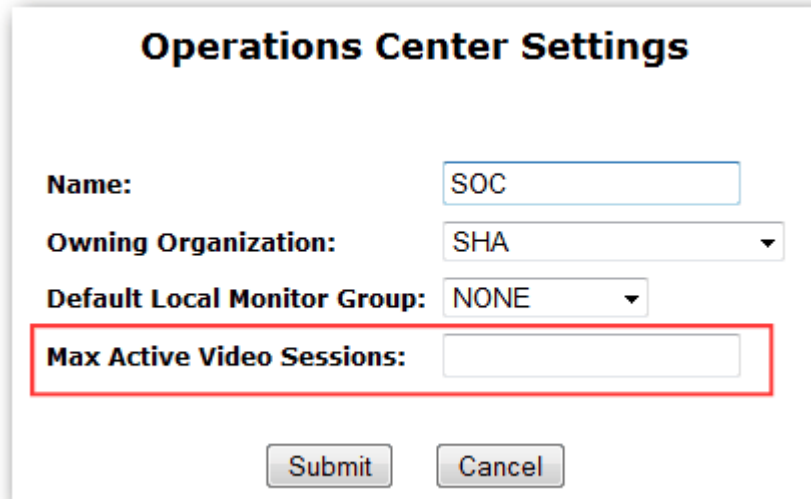
**Backup Operations Centers**

RITIS  
SOC  
TOC3

**Figure 7-16 Add Operations Center**

### 7.1.13 Operations Center Settings

The Operations Center Settings page is being modified in R9 to add the Max Active Video Sessions field, as described above.



The image shows a web form titled "Operations Center Settings". It contains four fields: "Name:" with a text input containing "SOC"; "Owning Organization:" with a dropdown menu showing "SHA"; "Default Local Monitor Group:" with a dropdown menu showing "NONE"; and "Max Active Video Sessions:" with a text input. The "Max Active Video Sessions:" field is highlighted with a red rectangular border. At the bottom of the form are two buttons: "Submit" and "Cancel".

<b>Name:</b>	SOC
<b>Owning Organization:</b>	SHA
<b>Default Local Monitor Group:</b>	NONE
<b>Max Active Video Sessions:</b>	

Submit Cancel

**Figure 7-17 Operations Center Settings**

## 7.2 System Interfaces

### 7.2.1 Class Diagrams

#### 7.2.1.1 ResourceManagement (Class Diagram)

This class diagram contains the interfaces pertaining to shared resources, operations centers, user login sessions, and organizations.

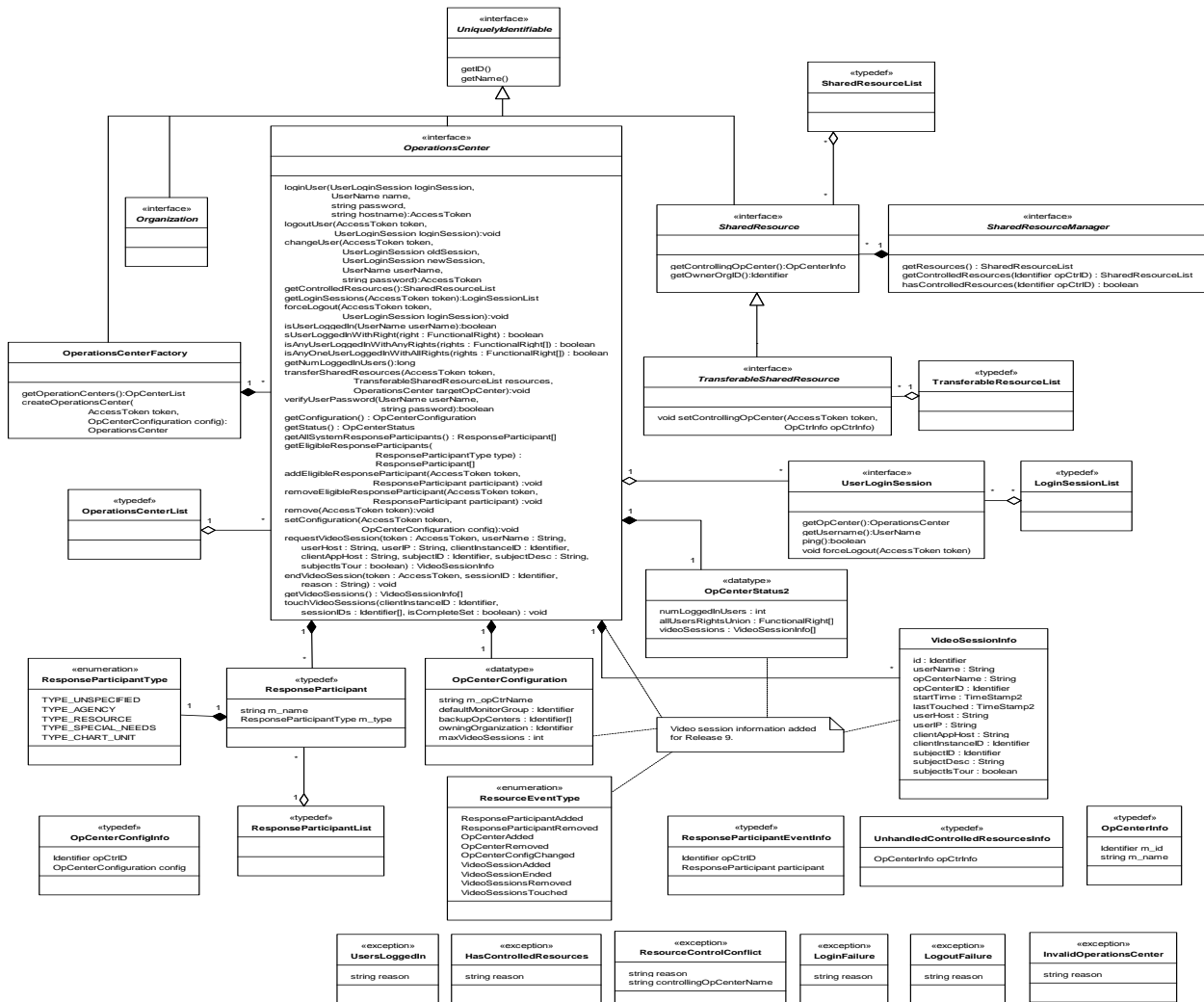


Figure 7-18. ResourceManagement (Class Diagram)

#### **7.2.1.1.1 HasControlledResources (Class)**

This class represents an exception which describes a failure caused when the user tries to do something which requires that no resources be controlled, yet the Operations Center which the user is logged in to is still controlling one or more shared resources.

#### **7.2.1.1.2 InvalidOperationsCenter (Class)**

Exception which describes a failure caused when the operations center specified is not valid for the attempted operation.

#### **7.2.1.1.3 LoginFailure (Class)**

This class represents an exception which describes a login failure.

#### **7.2.1.1.4 LoginSessionList (Class)**

A LoginSessionList is simply a collection of UserLoginSession objects.

#### **7.2.1.1.5 LogoutFailure (Class)**

This exception is thrown when an error occurs while logging a user out of the system.

#### **7.2.1.1.6 OpCenterConfigInfo (Class)**

This structure contains information pertaining to a change in the configuration of an operations center.

#### **7.2.1.1.7 OpCenterConfiguration (Class)**

This structure contains the configuration data for an operations center.

#### **7.2.1.1.8 OpCenterInfo (Class)**

This structure contains the information about an OperationsCenter.

#### **7.2.1.1.9 OpCenterStatus2 (Class)**

The actual name of this class is OpCenterStatus. It represents the status of an operations center. This class was introduced for R3B1 to transmit status of operations centers from the Resource Manager serving it to other interested parties. The data stored in the OpCenterStatus includes the number of users currently logged into the center and the union of all functional rights held by all users currently logged into that center.

#### **7.2.1.1.10 OperationsCenter (Class)**

The OperationsCenter represents a center where one or more users are located. This class is used to log users into the system. If the username and password provided to the loginUser method are valid, the caller is given a token that contains information about the user and the functional rights of the user. This token is then used to call privileged methods within the system. Shared resources in the system are either available or under the control of an OperationsCenter. The OperationsCenter keeps track of users that are logged in so that it can ensure that the last user does not log out while there are shared resources under its control. This list of logged in users is also available for monitoring system usage or to force users to logout for system maintenance.



#### **7.2.1.1.11                      OperationsCenterFactory (Class)**

This class is used to create new operations centers and maintain them in a collection.

#### **7.2.1.1.12                      OperationsCenterList (Class)**

This represents a collection of OperationsCenter objects.

#### **7.2.1.1.13                      Organization (Class)**

The Organization interface extends the UniquelyIdentifiable interface and will represent an organization, that is an administrative body which can control or own resources.

#### **7.2.1.1.14                      ResourceControlConflict (Class)**

This exception is thrown when attempt to gain control of a shared resource fails because the resource is under the control of a different operations center and the requesting user does not have the functional right to override the restriction.

#### **7.2.1.1.15                      ResourceEventType (Class)**

The ResourceEventType enumeration defines all of the resource related event types.

#### **7.2.1.1.16                      ResponseParticipant (Class)**

The ResponseParticipant class is a non-behavioral structure which specifies a participant in a response.

#### **7.2.1.1.17                      ResponseParticipantEventInfo (Class)**

This structure contains information about an eligible response participant that is added to an operations center.

#### **7.2.1.1.18                      ResponseParticipantList (Class)**

This represents a collection of ResponseParticipant objects.

#### **7.2.1.1.19                      ResponseParticipantType (Class)**

The ResponseParticipantType enumeration defines a type of entity participating in a response to an event. This could be an external organization, a mobile unit, a mobile device or special purpose vehicle, or a special needs vehicle equipped to handle unusual or hazardous situations.

#### **7.2.1.1.20                      SharedResource (Class)**

The SharedResource interface is implemented by any object that may have an operations center responsible for the disposition of the resource while the resource is in use.

#### **7.2.1.1.21                      SharedResourceList (Class)**

A SharedResourceList is simply a collection of SharedResource objects.

#### **7.2.1.1.22                      SharedResourceManager (Class)**

The SharedResourceManager interface is implemented by classes that manage shared resources. Implementing classes must be able to provide a list of all shared resources under their management. Implementing classes must also be able to tell others if there are any

resources under its management that are controlled by a given operations center. The shared resource manager is also responsible for periodically monitoring its shared resources to detect if the operations center controlling a resource doesn't have at least one user logged into the system. When this condition is detected, the shared resource manager must push an event on the ResourceManagement event channel to notify others of this condition.

#### **7.2.1.1.23                      TransferableResourceList (Class)**

This represents a collection of transferable shared resources.

#### **7.2.1.1.24                      TransferableSharedResource (Class)**

The TransferrableSharedResource interface extends the SharedResource interface, which is implemented by SharedResource objects whose control can be transferred from one operations center to another.

#### **7.2.1.1.25                      UnhandledControlledResourcesInfo (Class)**

The UnhandledControlledResourcesEvent class is an event pushed when it is detected that an OperationsCenter is controlling one or more controlled resources but has no users logged in.

#### **7.2.1.1.26                      UniquelyIdentifiable (Class)**

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

#### **7.2.1.1.27                      UserLoginSession (Class)**

The UserLoginSession CORBA interface is used to store information about a user that is logged into the system. This object is served from the GUI and provides a means for the servers to call back into the GUI process.

#### **7.2.1.1.28                      UsersLoggedIn (Class)**

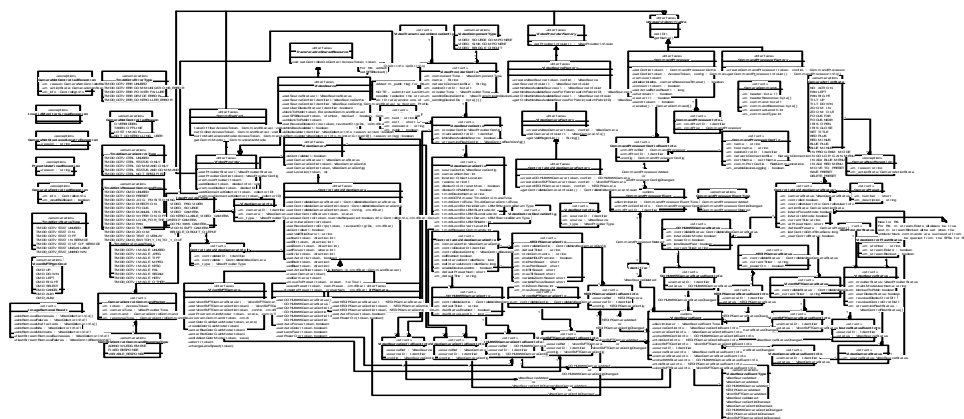
This exception is thrown if an attempt is made to remove the operations center when users are logged in.

#### **7.2.1.1.29                      VideoSessionInfo (Class)**

This structure represents information used to keep track of an active video session (i.e., an instance of video on the desktop).

### 7.2.1.2 CameraControlIDLClasses (Class Diagram)

This class diagram shows IDL generated classes used for defining video camera configuration in chart.



**Figure 7-19. CameraControlIDLClasses (Class Diagram)**

#### 7.2.1.2.1 CameraActionState (Class)

This enumeration identifies what action the camera is currently performing (if any).

#### 7.2.1.2.2 CameraBusyException (Class)

This exception is thrown if an attempt to issue an immediate mode camera control command (such as pan, tilt, etc.) is issued while the camera is performing a long-running command (such as a moveToPresetCommand or a setTitleCommand). This indicates to the operator that the camera is momentarily busy, and the operator should try the action again in a few seconds, or when the camera image on the monitor shows that the long-running request has completed.

#### 7.2.1.2.3 CameraCommand (Class)

CameraCommand contains information about the commands sent to, and responses received from, the camera.

#### 7.2.1.2.4 CameraControlCommandPacket (Class)

#### 7.2.1.2.5 CameraControlCommandResponseType (Class)

Identifies the camera control command response type expected from a command.

#### 7.2.1.2.6 CameraIsControlledException (Class)

This exception is used to indicate a request to control a camera has failed because it is already controlled, or a request to override control has failed because the requester does not have authority to override the current control session, or a request to move a camera to a preset for a tour has been denied because the camera is currently being controlled.

#### **7.2.1.2.7 CameraNotControlledException (Class)**

This is an exception thrown if an attempt to issue a camera control command is issued when the camera is not currently controlled by the requester. This is most likely to occur immediately after a control override, in cases where the client has not received or processed the override event yet.

#### **7.2.1.2.8 CameraNotControlledReason (Class)**

This enumeration identifies reasons why a CameraNotControlledException would be thrown.

#### **7.2.1.2.9 CameraNotDisplayedLocallyException (Class)**

This exception is thrown when a user tries to request control of a camera without having the camera displayed on a local monitor.

#### **7.2.1.2.10 CameraPreset (Class)**

This structure stores information about a preset configured for a camera.

#### **7.2.1.2.11 COHU3955Camera (Class)**

The COHUCamera interface is implemented by objects representing COHU-brand video cameras. The COHUCamera interface is extended by the COHUMPCCamera and COHU3955Camera interfaces. The COHUCamera interface includes all methods which are common to the two COHU cameras used by CHART II, the COHU MPC camera and the COHU 3955 camera. (Note that this interface may well contain a superset of methods which would be implemented by the entire line of all models of COHU video cameras).

#### **7.2.1.2.12 COHU3955CameraConfig (Class)**

This structure defines configuration data for the COHU 3955 model video camera.

#### **7.2.1.2.13 COHU3955CameraConfigEventInfo (Class)**

This struct is used for passing event data related to a COHU 3955 camera configuration. This is used when a COHU 3955 camera is added to the system or undergoes a configuration change. NOTE: The current configuration is passed along with object on adds as well as configuration changes, otherwise clients would immediately need to immediately query a new object for its configuration data after notification of the camera being added.

#### **7.2.1.2.14 COHU3955CameraStatus (Class)**

The CameraStatus class is an abstract value-type class which provides status information for a Camera. This status information is relatively dynamic: things like the communication mode, operational status, operation center information, status change time.

#### **7.2.1.2.15 COHU3955CameraStatusEventInfo (Class)**

This struct is used for passing event data related to a COHU 3955 camera status. This is used when a COHU 3955 video source undergoes a status change.

#### **7.2.1.2.16                    CommandProcessor (Class)**

The CommandProcessor interface is implemented by a class representing a command processor control port with direct connection to the control port of several video cameras. It is used to send video camera control commands and return responses to a camera control process.

#### **7.2.1.2.17                    CommandProcessorConfig (Class)**

This structure defines configuration data for a command processor.

#### **7.2.1.2.18                    CommandProcessorConfigEventInfo (Class)**

This struct is used for passing event data related to a CommandProcessor configuration when a CommandProcessor undergoes a configuration change.

#### **7.2.1.2.19                    CommandProcessorEvent (Class)**

This union identifies the data to be passed with CommandProcessor events that are pushed through the event service. The data pushed with these events is defined in the CommandProcessorEvent union.

#### **7.2.1.2.20                    CommandProcessorEventType (Class)**

This enum lists the events related to CommandProcessor control, which are pushed on a CommandProcessor event channel through the CORBA event service. The data pushed with these events is defined in the CommandProcessorEvent union.

#### **7.2.1.2.21                    CommandProcessorFactory (Class)**

This interface defines an object that is used to manage command processor objects in the system.

#### **7.2.1.2.22                    CommandProcessorInfo (Class)**

A structure of related information about a single CommandProcessor.

#### **7.2.1.2.23                    CommEnabled (Class)**

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

#### **7.2.1.2.24                    ControllableVideoCamera (Class)**

The ControllableVideoCamera interface is implemented by objects representing controllable video cameras within the CHART II system. The ControllableVideoCamera interface represents a controllable video camera as opposed to the uncontrollable, immovable VideoCamera. Current plans call for classes to represent a COHU MPC

camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera, and there are interfaces defined for each of these subtypes of ControllableVideoCamera. The ControllableVideoCamera interface includes all methods common to the three known types of video cameras currently in use by MDSHA, although it is likely to contain a superset of methods which would be implemented by the entire universe of all video cameras which could someday be used. This interface may have to be refined in the event that future brands or models of video cameras might be incorporated under CHART II, but it is an appropriate set of methods for the present day. Current plans call for classes to represent a COHU MPC camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera.

#### **7.2.1.2.25 ControllableVideoCameraConfig (Class)**

The ControllableVideoCameraConfig is used to hold and transmit configuration information about ControllableVideoCamera objects at the ControllableVideoCamera level.

#### **7.2.1.2.26 ControllableVideoCameraFactory (Class)**

#### **7.2.1.2.27 ControllableVideoCameraInfo (Class)**

A structure of related information about a single ControllableVideoCamera.

#### **7.2.1.2.28 ControllableVideoCameraStatus (Class)**

The ControllableVideoCameraStatus is used to hold and transmit status information about ControllableVideoCameraStatus objects at the ControllableVideoCamera level.

#### **7.2.1.2.29 Identifier (Class)**

Wrapper class for a CHART2 identifier byte sequence. This class will be used to add identifiable objects to hash tables and perform subsequent lookup operations.

#### **7.2.1.2.30 ImageRemovalResult (Class)**

This structure contains the results of a call to either of the VideoSource operations blockToPublic() or revokeDisplay(). This structure contains lists comprising all VideoSink objects which were displaying this image (or were believed to be) giving the status of each. This list also contains flash stream configurations which contain information for where this VideoSource is streaming.

#### **7.2.1.2.31 InvalidMonitorGroupException (Class)**

This exception is used to indicate a request to control a camera has failed because the MonitorGroup provided in the ControllingInfo in the request is not known to exist.

#### **7.2.1.2.32 NTCIPCamera (Class)**

This interface is used to represent an NTCIP model video camera in the field. The system contains an instance of this interface for each NTCIP video camera.

#### **7.2.1.2.33 NTCIPCameraConfig (Class)**

This structure defines configuration data for the NTCIP type video camera.

#### **7.2.1.2.34 NTCIPCameraConfigEventInfo (Class)**

This struct is used for passing event data related to a NTCIP camera configuration. This is used when a NTCIP camera is added to the system or undergoes a configuration change.

NOTE: The current configuration is passed along with object on adds as well as configuration changes, otherwise clients would immediately need to immediately query a new object for its configuration data after notification of the camera being added.

#### **7.2.1.2.35 NTCIPCameraStatus (Class)**

This structure defines the status data for the NTCIP video camera type.

#### **7.2.1.2.36 NTCIPCameraStatusEventInfo (Class)**

This struct is used for passing event data related to a NTCIP camera status. This is used when a NTCIP video camera undergoes a status change.

#### **7.2.1.2.37 PresetUndefinedException (Class)**

This exception is used to indicate a moveToPreset request has specified an undefined preset.

#### **7.2.1.2.38 TmddCameraControlType (Class)**

#### **7.2.1.2.39 TmddCctvErrorType (Class)**

#### **7.2.1.2.40 TmddCctvImageType (Class)**

This enum lists the values that can be used to describe a type of camera image using the specific TMDD-prescribed values for cctv\_image (Reference TMDD Vol II Annex June 2004).

#### **7.2.1.2.41 TmddCctvRequestCommandType (Class)**

#### **7.2.1.2.42 TmddCctvStatusType (Class)**

#### **7.2.1.2.43 TransferableSharedResource (Class)**

The TransferrableSharedResource interface extends the SharedResource interface, which is implemented by SharedResource objects whose control can be transferred from one operations center to another.

#### **7.2.1.2.44 UniquelyIdentifiable (Class)**

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

#### **7.2.1.2.45 ViconSVFTCamera (Class)**

This interface is used to represent a Vicon Surveyor VFT model video camera in the field. The system contains an instance of this interface for each Vicon SVFT video camera.

#### **7.2.1.2.46 ViconSVFTCameraConfig (Class)**

This structure defines configuration data for the Vicon Surveyor VFT model video camera. At present time, this structure adds nothing to the "base class" ControllableVideoCameraConfig structure.

#### **7.2.1.2.47                      ViconSVFTCameraConfigEventInfo (Class)**

This struct is used for passing event data related to a Vicon SVFT camera configuration. This is used when a Vicon SVFT camera is added to the system or undergoes a configuration change. NOTE: The current configuration is passed along with object on adds as well as configuration changes, otherwise clients would immediately need to immediately query a new object for its configuration data after notification of the camera being added.

#### **7.2.1.2.48                      ViconSVFTCameraStatus (Class)**

The ViconSVFTCameraStatus class is used to hold camera status information at the ViconSVFTCamera level. Only ViconSVFTCamera specific information is stored.

#### **7.2.1.2.49                      ViconSVFTCameraStatusEventInfo (Class)**

This struct is used for passing event data related to a Vicon SVFT camera status. This is used when a Vicon SVFT video source undergoes a status change.

#### **7.2.1.2.50                      ViconSVFTPgmCmd (Class)**

This enumeration defines the program commands that can be sent to the Vicon SVFT camera while it is in program mode. (Some of these commands can also be sent while in various SVFT color gain menu layers.)

#### **7.2.1.2.51                      VideoCamera (Class)**

The VideoCamera interface is implemented by objects representing controllable video cameras within the CHART II system. The VideoCamera interface represents a controllable video camera as opposed to the uncontrollable, immovable FixedVideoCamera, the other type of GenericVideoCamera. (The VideoCamera class could have been called the ControllableVideoCamera interface, but since the CHART II video system exists primarily to control controllable video cameras, the camera hierarchy has been arranged to avoid the longish name ControllableVideoCamera.) Current plans call for classes to represent a COHU MPC camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera, and there are interfaces defined for each of these subtypes of VideoCamera. The VideoCamera interface includes the GeoLocatable interface, to someday allow for advanced features such as automatic identification of cameras near traffic events, automatic pointing of cameras to traffic events, etc.

The VideoCamera interface includes all methods common to the three known types of video cameras currently in use by MDSHA, although it is likely to contain a superset of methods which would be implemented by the entire universe of all video cameras which could someday be used. This interface may have to be refined in the event that future brands or models of video cameras might be incorporated under CHART II, but it is an appropriate set of methods for the present day.

#### **7.2.1.2.52                      VideoCameraConfig (Class)**

The VideoCameraConfig structure is used to hold configuration information about VideoCamera objects at the VideoCamera level. Further details about lower-level VideoCamera subclasses are provided by subclasses of VideoCameraConfig.



#### **7.2.1.2.53 VideoCameraConfigEventInfo (Class)**

This struct is used for passing event data related to a video camera configuration. This is used when a fixed video camera is added to the system or undergoes a configuration change. The video camera in this struct is nothing more than a video camera. NOTE: The current configuration is passed along with object on adds as well as configuration changes, otherwise clients would immediately need to immediately query a new object for its configuration data after notification of the camera being added.

#### **7.2.1.2.54 VideoCameraFactory (Class)**

The VideoCameraFactory interface is implemented by factory classes responsible for creating, maintaining, and controlling a collection of VideoCamera objects.

#### **7.2.1.2.55 VideoCameraInfo (Class)**

#### **7.2.1.2.56 VideoCameraStatus (Class)**

The VideoCameraStatus structure is used to hold status information about VideoCamera objects at the VideoCamera level. Further details about lower-level VideoCamera subclasses are provided by subclasses of VideoCameraStatus.

#### **7.2.1.2.57 VideoCameraStatusEventInfo (Class)**

This struct is used for passing event data related to a video source status. This is used when a fixed video camera undergoes a status change.

#### **7.2.1.2.58 VideoComponentType (Class)**

This enum lists the video component types supported by the software.

#### **7.2.1.2.59 VideoControlDeviceConfig (Class)**

This structure stores configuration information used to find and use the video control device used to send/receive camera control commands/responses to/from a camera.

#### **7.2.1.2.60 VideoControlFlashConfig (Class)**

This structure stores configuration information about a flash streaming server configuration that is displaying a camera's image.

#### **7.2.1.2.61 VideoControlFlashStatus (Class)**

This structure contains information about the existence and blocked status of a video source's stream within a Streaming Flash Server (SFS).

#### **7.2.1.2.62 VideoProvider (Class)**

The VideoProvider interface is a generic abstract interface including VideoSource objects (e.g. video cameras) and BridgeCircuit objects. Both VideoSource and BridgeCircuit objects provide video to a VideoCollector, but only VideoSource objects are true origins of video which a typical user would have direct interaction with. BridgeCircuit VideoProvider objects merely pass on video provided from elsewhere in a VideoRoute.

#### **7.2.1.2.63                      VideoProviderConfig (Class)**

This structure defines configuration data common to all video sources.

#### **7.2.1.2.64                      VideoProviderFactory (Class)**

This interface defines an object that is used to manage video provider objects in the system. There is no create operation because VideoProvider is an abstract interface.

#### **7.2.1.2.65                      VideoProviderStatus (Class)**

The VideoProviderStatus structure is used to hold and transmit status information about VideoProvider objects at the VideoProvider level. Further details about lower-level VideoProvider subclasses are provided by subclasses of VideoProviderStatus.

#### **7.2.1.2.66                      VideoProviderType (Class)**

This enum lists the different types of VideoProvider in the system.

#### **7.2.1.2.67                      VideoSource (Class)**

The VideoSource interface is implemented by objects which originate video signals, such as video cameras and image generators. Within the user interface, the VideoSource interface represents all video sources which can be put on monitors (i.e., VideoSink objects).

The VideoSource interface includes the SharedResource interface. A VideoSource is controlled by an Operations Center if the VideoSource is in maintenance mode, or if the VideoSource is a camera which has an active control session up.

#### **7.2.1.2.68                      VideoSourceConfig (Class)**

This structure defines configuration data common to all video sources.

#### **7.2.1.2.69                      VideoSourceConfigEventInfo (Class)**

This struct is used for passing event data related to a video source configuration. This is used when a generic video source is added to the system or undergoes a configuration change. The video source in this struct is nothing more than a video source. NOTE: The current configuration is passed along with object on adds as well as configuration changes, otherwise clients would immediately need to immediately query a new object for its configuration data after notification of the source being added.

#### **7.2.1.2.70                      VideoSourceEvent (Class)**

This union identifies the data to be passed with video source events that are pushed through the event service.

#### **7.2.1.2.71                      VideoSourceEventType (Class)**

This enum lists the events related to camera control that are pushed on a camera event channel through the CORBA event service. The data pushed with these events is defined in the VideoSourceEvent union.

#### **7.2.1.2.72                      VideoSourceFactory (Class)**

This interface defines an object that is used to manage the creation of source objects in the system.

#### **7.2.1.2.73                      VideoSourceInfo (Class)**

A struct of related information about a single VideoSource.

#### **7.2.1.2.74                      VideoSourceStatus (Class)**

The VideoSourceStatus structure is used to hold and transmit status information about VideoSource objects at the VideoSource level. Further details about lower-level VideoSource subclasses are provided by subclasses of VideoSourceStatus.

#### **7.2.1.2.75                      VideoSourceStatusEventInfo (Class)**

This struct is used for passing event data related to a video source status. This is used when a generic video source undergoes a status change.

#### **7.2.1.2.76                      VideoTransmissionDeviceConfig (Class)**

This structure defines configuration data common to all video transmission devices.

### 7.3.1.1 ResourceClasses (Class Diagram)

[illegible]

### Figure 7-20. ResourceClasses (Class Diagram)

#### 7.3.1.1.1 java.util. TimerTask (Class)

### 7.3.1.1.2 OpCenterConfiguration (Class)

### 7.3.1.1.3 OpCenterStatus2 (Class)

---

08/26/2011

OpCenterStatus includes the number of users currently logged into the center and the union of all functional rights held by all users currently logged into that center.

#### **7.3.1.1.4                      OperationsCenter (Class)**

The OperationsCenter represents a center where one or more users are located. This class is used to log users into the system. If the username and password provided to the loginUser method are valid, the caller is given a token that contains information about the user and the functional rights of the user. This token is then used to call privileged methods within the system. Shared resources in the system are either available or under the control of an OperationsCenter. The OperationsCenter keeps track of users that are logged in so that it can ensure that the last user does not log out while there are shared resources under its control. This list of logged in users is also available for monitoring system usage or to force users to logout for system maintenance.

#### **7.3.1.1.5                      OperationsCenterFactory (Class)**

This class is used to create new operations centers and maintain them in a collection.

#### **7.3.1.1.6                      OperationsCenterFactoryImpl (Class)**

This class provides implementation of OperationsCenterFactory interface to manage OperationCenter objects in the system.

#### **7.3.1.1.7                      OperationsCenterImpl (Class)**

This class provides the implementation of the OperationsCenter interface for this module. It, therefore, provides a concrete implementation of each of the methods in the interface. It also contains a collection of UserLoginSession objects, one for each user who is currently logged in.

#### **7.3.1.1.8                      Organization (Class)**

The Organization interface extends the UniquelyIdentifiable interface and will represent an organization, that is an administrative body which can control or own resources.

#### **7.3.1.1.9                      OrganizationImpl (Class)**

This class provides the implementation of the Organization interface for this module. Thus, it provides a concrete implementation of each of the methods in the interface.

#### **7.3.1.1.10                      ProxySimpleOpCenter (Class)**

This class is used as a proxy for operations centers existing in all user management services (including the local service). The proxy operations centers cached are not complete copies of the operations centers, because the full range of data is not needed. The ProxySimpleOpCenter data consists of the OpCenterConfiguration and the OpCenterStatus, but not the center's participant data. (This is why the names of this object contains the word "Simple".) These proxy operations centers allow every alert module service in the system to have some knowledge of every operations center in the entire system, for the quickly determining rights of the users at those operations centers.

#### **7.3.1.1.11 ResourcesDB (Class)**

This class provides a set of API calls to access the Operations Center data from the database. The API's provide functionality to add, remove and retrieve Operation Center data from the database. The connection to the database is acquired from the Database object which manages all the database connections.

#### **7.3.1.1.12 ResourcesModule (Class)**

This module creates, publishes and destroys all objects related to resource management that are used by the User Management service application.

#### **7.3.1.1.13 ResourcesModuleProperties (Class)**

This class contains settings used by the Resources Module classes. These settings are read and initialized at startup and do not change while the application is running..

#### **7.3.1.1.14 ServiceApplication (Class)**

This interface is implemented by objects that can provide the basic services needed by a ChartII service application. These services include providing access to basic CORBA objects that are needed by service applications, such as the ORB, POA, Trader, and Event Service.

#### **7.3.1.1.15 ServiceApplicationModule (Class)**

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

#### **7.3.1.1.16 UserLoginSession (Class)**

The UserLoginSession CORBA interface is used to store information about a user that is logged into the system. This object is served from the GUI and provides a means for the servers to call back into the GUI process.

#### **7.3.1.1.17 UserManagementDB (Class)**

The UserManagementDB Class provides methods used to access and modify User Managment data in the database. This class uses a Database object to retrieve a connection to the database for its exclusive use during a method call.

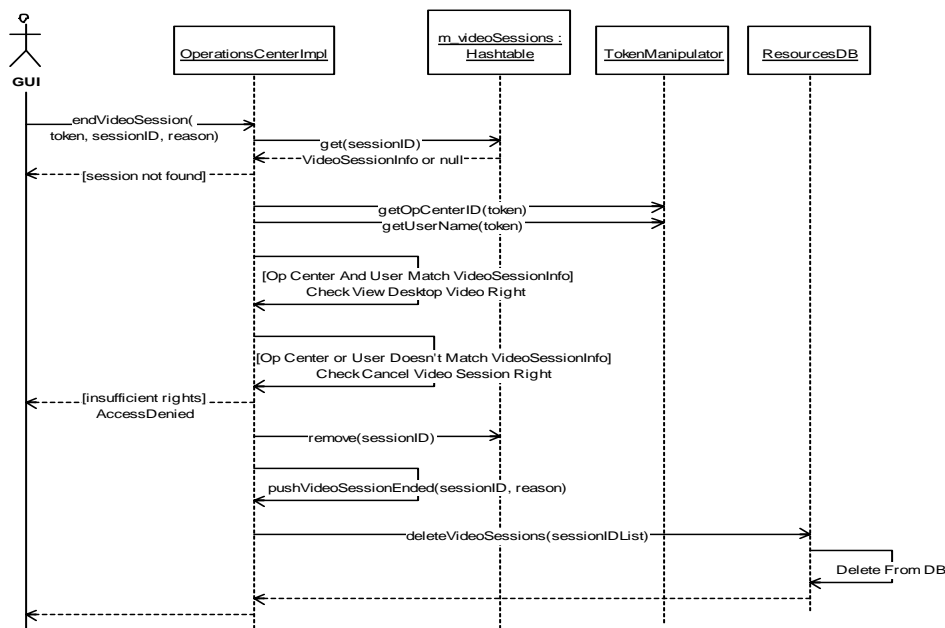
#### **7.3.1.1.18 VideoSessionCleanupTask (Class)**

This class is executed on a timer to clean up any old video sessions that have not been used in a long enough time.

## 7.3.2 Sequence Diagrams

### 7.3.2.1 OperationsCenterImpl:endVideoSession (Sequence Diagram)

This diagram shows the processing when a call is made to end a video session, either by a user ending his/her own session (when closing a video window) or by an administrator ending the video session of another user. The VideoSessionInfo struct is retrieved from the OperationCenterImpl's lookup table. The operations center and user name are extracted from the user's access token, and are checked against the VideoSessionInfo to see if they match the op center and user name for the video session. If they match, the View Desktop Video functional right is checked; otherwise, the Cancel Video Session right is checked. If the user has insufficient rights, an AccessDenied exception is thrown; otherwise, the session is removed from the OperationCenterImpl's table of video sessions, a CORBA event is pushed indicating the session ID and the reason for ending it, and the session is deleted from the database.

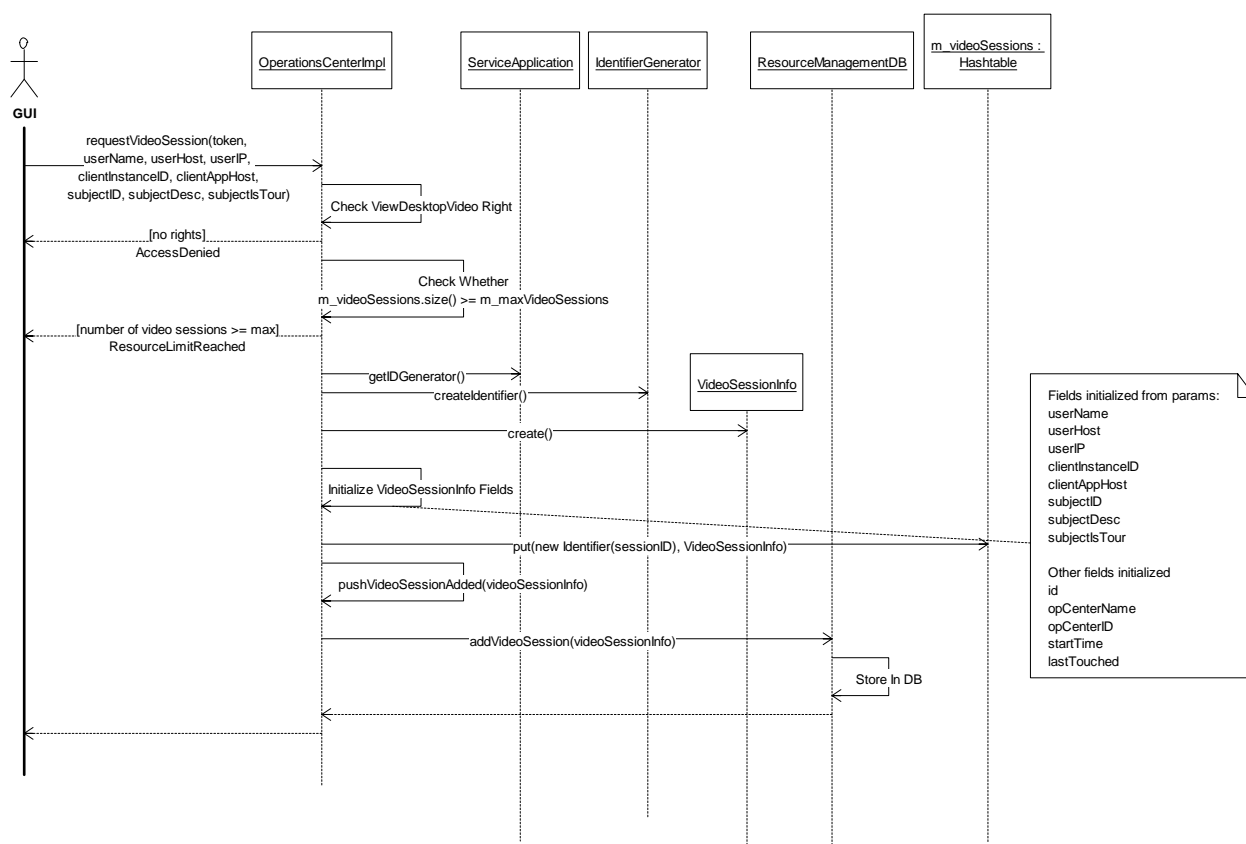


**Figure 7-21. OperationsCenterImpl:endVideoSession (Sequence Diagram)**

### 7.3.2.2

### OperationsCenterImpl:requestVideoSession (Sequence Diagram)

This diagram shows the processing for requesting a video session. The GUI calls requestVideoSession(), which checks the user's token for the View Desktop Video right and throws an exception if not found. Then it checks whether the number of video sessions managed by the operations center is already at (or over) the maximum number allowed for the operations center. If the limit has been reached an exception is thrown; otherwise, a new VideoSessionInfo object is created and initialized. The video session info is added to the operations center's table of video sessions, and a VideoSessionAdded CORBA event is pushed. The video session data is added to the database, to persist the video session in case the service is restarted.



**Figure 7-22. OperationsCenterImpl:requestVideoSession (Sequence Diagram)**



### 7.3.2.3

### OperationsCenterImpl:touchVideoSessions (Sequence Diagram)

This diagram shows the processing when the client application calls to "touch" the video sessions it is managing, to declare that they are still in use. When touchVideoSessions() is called, for each session ID the VideoSessionInfo is found within the OperationsCenterImpl's table, and the "lastTouched" timestamp is updated in the session info. The session info is added to a list of sessions updated, for later use. Next, if the isCompleteSet parameter is true, the OperationsCenterImpl iterates through all of its known sessions and if the clientInstanceID matches the video sessions client instance ID but the lastTouched timestamp is before the current time, the video session ID is added to the list to remove. (The timestamp not being updated indicates that the session is no longer managed by the client application, if the isCompleteSet parameter was true). CORBA events are pushed for any updated or removed sessions. The changes are stored in the database as a best effort to retain the state of video sessions if the service is restarted (although since the video sessions are transient in nature, persisting the sessions is not critical).

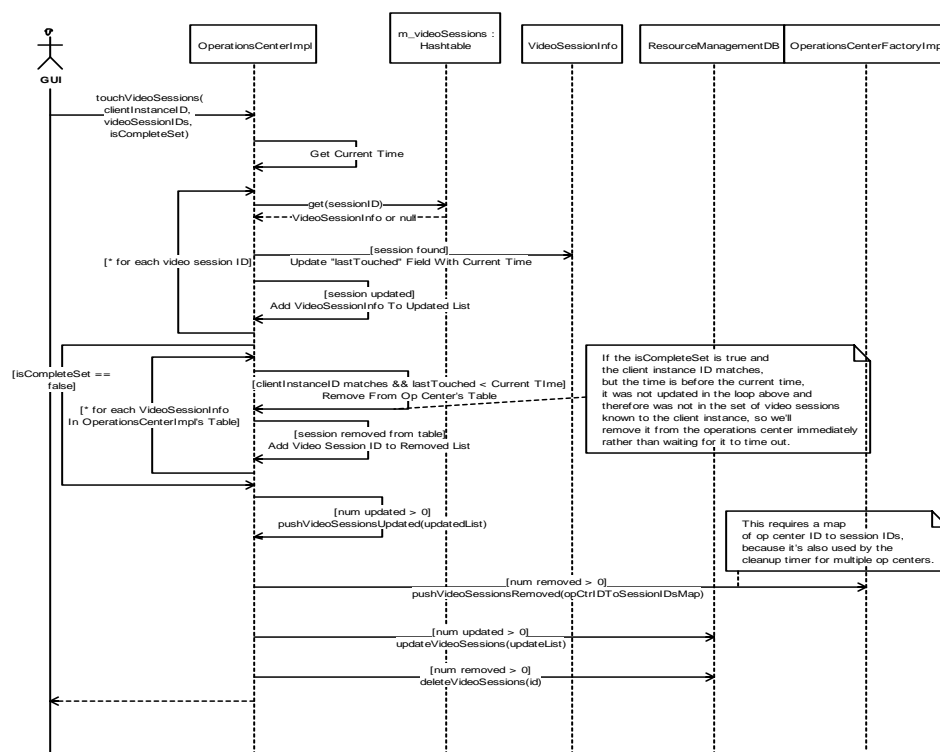
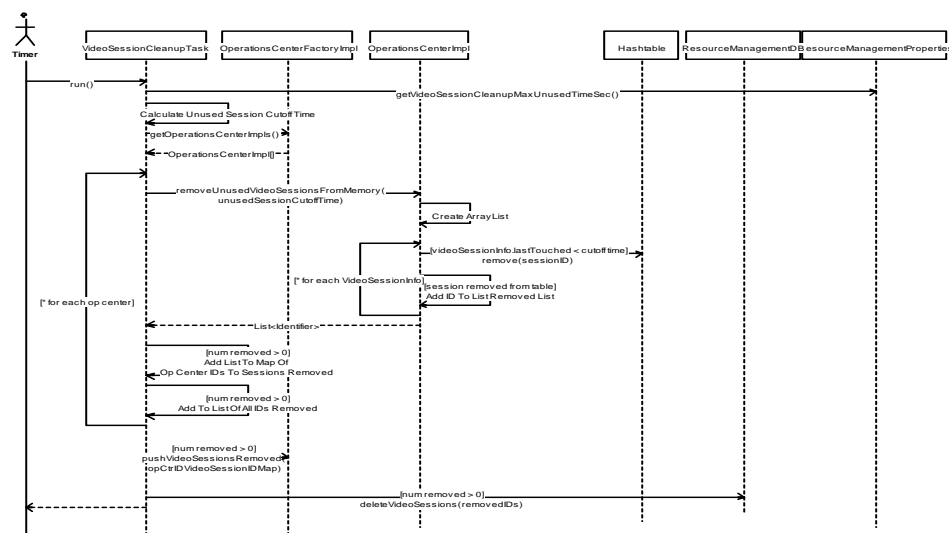


Figure 7-23. OperationsCenterImpl:touchVideoSessions (Sequence Diagram)

### 7.3.2.4 VideoSessionCleanupTask:run (Sequence Diagram)

This shows the processing to clean up unused video sessions, such as could be left stranded when a GUI is shut down or crashes. First the cutoff time is calculated using the maximum unused session age. Each OperationsCenterImpl is called to remove its unused video sessions from memory if their "last touched" time is older than the cutoff time. After removing them from memory, the list of removed sessions is returned. A CORBA event is then pushed containing the op center ID and the list of session IDs for each op center with sessions removed. The video sessions are then deleted from the database.



**Figure 7-24. VideoSessionCleanupTask:run (Sequence Diagram)**

## 7.4 CameraControlModule

### 7.4.1 State Diagram

#### 7.4.1.1 CameraControlModule (State Diagram)

This diagram shows how a user can move between the camera viewing and camera control states using links available on various dialogs. Notice the diagram is symmetrical except for the lack of a link to move from the Desktop Video state to the Control Only state. This is because once you have a desktop video session, two actions are required to gain camera control and remove the video.

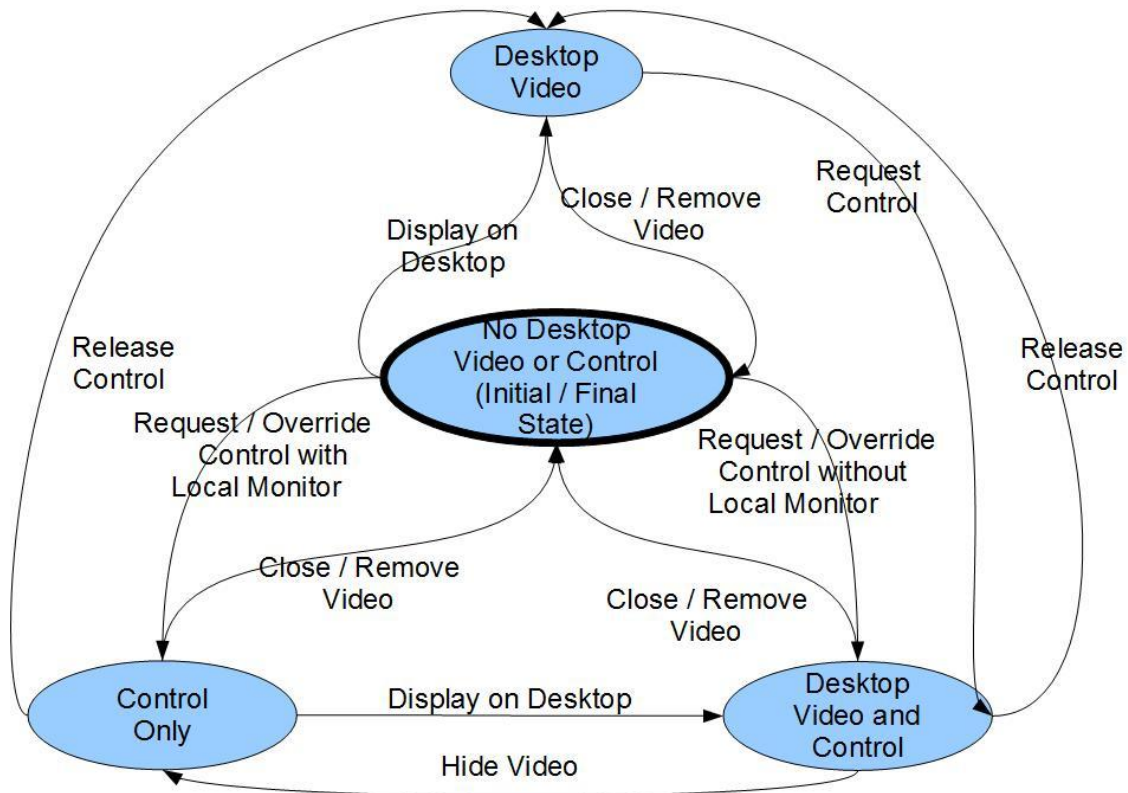
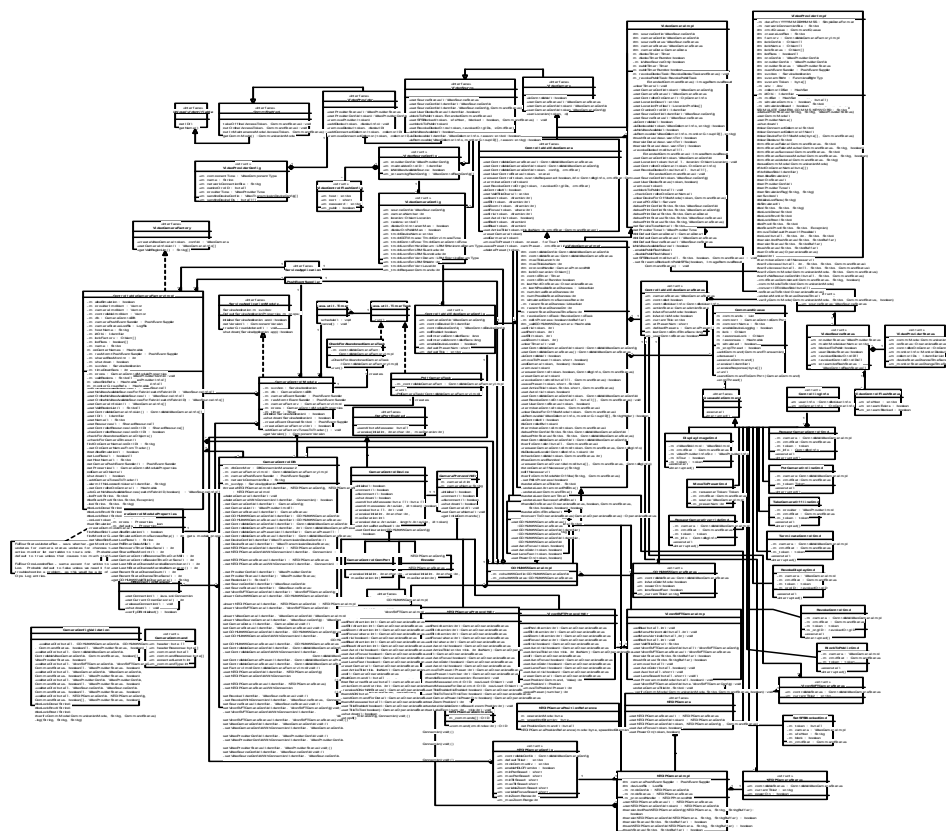


Figure 7-25. CameraControlModule (State Diagram)

## 7.4.2 Class Diagrams

### 7.4.2.1 CameraControlModule (Class Diagram)

This diagram shows the classes with comprise the CameraControlModule. The CameraControlModule is an installable module that serves the camera-type objects and factories to the rest of the CHART II system. This diagram shows how the implementation of these CORBA interfaces rely on other supporting classes to perform their functions. The CameraControlModule is responsible for serving all VideoSource objects including controllable cameras, fixed cameras, No Video Available sources, and potentially any other image generators, etc. The COHU3955CameraImpl, viconSVFTCameraImpl, and NTCIPCameraImpl are the primary classes operating in this module. These objects provide all access to the camera status and configuration. The CameraControlModule also includes factory implementations responsible for providing lists of cameras and other such objects to interested clients.



**Figure 7-26. CameraControlModule (Class Diagram)**

#### **7.4.2.1.1                   BlockToPublicCmd (Class)**

This class represents the information needed to create a block camera to public command to be added on the CommandQueue.

#### **7.4.2.1.2                   CameraCommand (Class)**

CameraCommand contains information about the commands sent to, and responses received from, the camera.

#### **7.4.2.1.3                   CameraConfigValidation (Class)**

This class provides validates camera configuration data for any type of camera (Video Source, (Fixed) Video Camera, COHU3955, SVFT, and NTCIP).

#### **7.4.2.1.4                   CameraControlComPort (Class)**

The CameraControlComPort interface is implemented by a class representing a COM port with direct connection to the control port of a video camera. It is used to send video camera control commands and return responses to a camera control process.

#### **7.4.2.1.5                   CameraControlDB (Class)**

The CameraControlDB class provides an interface between the Camera service and the database used to persist and depersist the Camera objects and their configuration and status in the database. It contains a collection of methods that perform database operations on tables pertinent to Camera Control. The class is constructed with a DBConnectionManager object, which manages database connections. Methods exist to insert and delete Camera objects from the database, and to get and set their configuration and status information.

#### **7.4.2.1.6                   CameraControlDevice (Class)**

The CameraControlDevice interface is implemented by classes which provide communications for access to control functions for a video camera. This includes encoders, command processors, and direct COM ports.

#### **7.4.2.1.7                   CameraControlModule (Class)**

The CameraControlModule class is the service module for the Camera devices and a Camera factory. It implements the ServiceApplicationModule interface. It creates and serves a single CameraFactoryImpl object, which in turn serves zero or more CameraImpl objects. It also creates CameraControlDB, CameraControlModuleProperties, and PushEventSupplier objects.

#### **7.4.2.1.8                   CameraControlModuleProperties (Class)**

The CameraControlModuleProperties class is used to provide access to properties used by the Camera Control Module. This class wraps properties that are passed to it upon construction. It adds its own defaults and provides methods to extract properties specific to the Camera Control Module.

#### **7.4.2.1.9                   CameraProtocolHdlr (Class)**

CameraProtocolHdlr classes provide implementations for all the camera commands. Each

CameraImpl class will have a CameraProtocolHdlr instantiated when initialized. When a camera control command is sent to the CameraImpl, CameraProtocolHdlr will be called to translate the command to byte messages which the camera understands. Then those messages are sent by the CameraControlDevice to the camera. CameraProtocolHdlr is capable of using different CameraControlDevice which is created during the initialization.

#### **7.4.2.1.10 CheckForAbandonedCameraTask (Class)**

The CheckForAbandonedCameraTask is a timer task. When the timer fires, it checks to see if a camera control session has exceeded the timeout, or whether a camera is controlled by an Operations center with no one logged in.

#### **7.4.2.1.11 COHU3955Camera (Class)**

The COHUCamera interface is implemented by objects representing COHU-brand video cameras. The COHUCamera interface is extended by the COHUMPPCamera and COHU3955Camera interfaces. The COHUCamera interface includes all methods which are common to the two COHU cameras used by CHART II, the COHU MPC camera and the COHU 3955 camera. (Note that this interface may well contain a superset of methods which would be implemented by the entire line of all models of COHU video cameras).

#### **7.4.2.1.12 COHU3955CameraImpl (Class)**

This class implements the COHU3955Camera interface, and inherits from the ControllableCameraImpl class. The COHU3955CameraImpl implements methods of COHU3955Camera, extending the controllable camera to include 3955-specific operations. This class will contain a configuration and status object as necessary to convey 3955-specific configuration and status information.

#### **7.4.2.1.13 COHU3955CameraStatus (Class)**

The CameraStatus class is an abstract value-type class which provides status information for a Camera. This status information is relatively dynamic: things like the communication mode, operational status, operation center information, status change time.

#### **7.4.2.1.14 COHUProtocolHdlr (Class)**

COHUProtocolHdlr is the base class for all COHU cameras. At present, this class contains implementations for common functions for COHU MPC and COHU 3955 cameras

#### **7.4.2.1.15 CommandProcessor (Class)**

The CommandProcessor interface is implemented by a class representing a command processor control port with direct connection to the control port of several video cameras. It is used to send video camera control commands and return responses to a camera control process.

#### **7.4.2.1.16 CommandQueue (Class)**

The CommandQueue class provides a queue for QueueableCommand objects. The CommandQueue has a thread that it uses to process each QueueableCommand in a first in first out order. As each command object is pulled off the queue by the CommandQueue's thread, the command object's execute method is called, at which time the command

performs its intended task.

#### **7.4.2.1.17                      CommEnabled (Class)**

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

#### **7.4.2.1.18                      ControllableCameraFactoryImpl (Class)**

The CameraFactoryImpl class provides an implementation of the CameraFactory interface (and its CameraFactory and SharedResourceManager interfaces) as specified in the IDL. The CameraFactoryImpl maintains a list of CameraImpl objects and is responsible for publishing Camera objects in the Trader on startup and as new camera objects are created. Whenever a Camera is created or removed, that information is persisted to the database. This class is also responsible for performing the checks requested by the timer tasks: to poll the Camera devices, to look for Camera devices with timeout exceeded, to look for Camera devices with no one logged in at the controlling operations center, and to initiate recovery processing as needed

#### **7.4.2.1.19                      ControllableVideoCamera (Class)**

The ControllableVideoCamera interface is implemented by objects representing controllable video cameras within the CHART II system. The ControllableVideoCamera interface represents a controllable video camera as opposed to the uncontrollable, immovable VideoCamera. Current plans call for classes to represent a COHU MPC camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera, and there are interfaces defined for each of these subtypes of ControllableVideoCamera. The ControllableVideoCamera interface includes all methods common to the three known types of video cameras currently in use by MDSHA, although it is likely to contain a superset of methods which would be implemented by the entire universe of all video cameras which could someday be used. This interface may have to be refined in the event that future brands or models of video cameras might be incorporated under CHART II, but it is an appropriate set of methods for the present day. Current plans call for classes to represent a COHU MPC camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera.

#### **7.4.2.1.20                      ControllableVideoCameraConfig (Class)**

The ControllableVideoCameraConfig is used to hold and transmit configuration information about ControllableVideoCamera objects at the ControllableVideoCamera level.

#### **7.4.2.1.21                      ControllableVideoCameraImpl (Class)**

The ControllableCameraImpl class provides an implementation of the ControllableVideoCamera interface and is derived from the CameraImpl class

implementing the VideoCamera interface.

This class contains a CommandQueue object that is used to sequentially execute long running operations related to camera control in a thread separate from the CORBA request threads, thus allowing quick initial responses.

Also contained in this class are ControllableVideoCameraConfig and ControllableVideoCameraStatus objects (used to store the configuration and status of the camera), and a VideoCameraData object (used to store internal status information which is persisted but not pushed out to clients).

The ControllableCameraImpl contains \*Impl methods that map to methods specified in the IDL, including requests to request control of the camera, terminate control of the camera, override control of the camera, and to send pan/tilt/zoom (PTZ) commands to the camera. Some of these requests are long running, so each request is stored in a specific subclass of QueueableCommand and added to the CommandQueue. The queueable command objects simply call the appropriate ControllableCameraImpl method as the command is executed by the CommandQueue in its thread of execution. PTZ commands are not considered long running and are not placed on the command queue.

The ControllableCameraImpl also contains methods called by the CameraFactory to support the timer tasks of the Camera Service: to poll the Camera, to look for Camera devices with communications timeout exceeded.

#### **7.4.2.1.22 ControllableVideoCameraStatus (Class)**

The ControllableVideoCameraStatus is used to hold and transmit status information about ControllableVideoCameraStatus objects at the ControllableVideoCamera level.

#### **7.4.2.1.23 ControllingInfo (Class)**

The ControllingInfo structure contains information about the entity controlling (or requesting to control) a VideoCamera.

#### **7.4.2.1.24 DataPortEnabled (Class)**

This interface is implemented by device specific communications classes. This interface provides an extra layer to remove dependencies on device specific packages.

#### **7.4.2.1.25 DBConnectionManager (Class)**

This class implements a database connection manager that manages a pool of database connections. Any CHART II system thread requiring database access gets a database connection from the pool of connections maintained by this manager class. The connections are maintained in two separate lists namely, inUseList and freeList. The inUseList contains connections that have already been assigned to a thread. The freeList contains unassigned connections. This class assumes that an appropriate JDBC driver has been loaded either by using the "jdbc.drivers" system property or by loading it explicitly. The class has a monitor thread that is started by the constructor. This connection monitor thread periodically checks the inUseList to see if there are connections that are owned by dead threads and move such connections to the freeList. The connection monitor thread is started only if a non-zero



value is specified for the monitoring time interval in the constructor.

#### **7.4.2.1.26                      DisplayImageCmd (Class)**

This class represents the information needed to create a display image command to be added on the CommandQueue.

#### **7.4.2.1.27                      Encoder (Class)**

The Encoder interface is implemented by classes representing any type of video encoder. The Encoder interface includes both the Codec and the VideoSendingDevice interfaces, which means in addition to providing forwarding of video, it also is used to send video camera control commands and return responses to a camera control process.

#### **7.4.2.1.28                      java.util.Timer (Class)**

This class provides asynchronous execution of tasks that are scheduled for one-time or recurring execution.

#### **7.4.2.1.29                      java.util.TimerTask (Class)**

This class is an abstract base class which can be scheduled with a timer to be executed one or more times.

#### **7.4.2.1.30                      MoveToPresetCmd (Class)**

This class represents the information needed to create a move to preset command to be added on the CommandQueue.

#### **7.4.2.1.31                      NTCIPCamera (Class)**

This interface is used to represent an NTCIP model video camera in the field. The system contains an instance of this interface for each NTCIP video camera.

#### **7.4.2.1.32                      NTCIPCameraCommands (Class)**

This class holds the ntcip command OIDs so the mib db does not have to be queried after startup.

#### **7.4.2.1.33                      NTCIPCameraConfig (Class)**

This structure defines configuration data for the NTCIP type video camera.

#### **7.4.2.1.34                      NTCIPCameraImpl (Class)**

This class implements the NTCIPCamera interface, and inherits from the ControllableCameraImpl class. The NTCIPCameraImpl implements methods of NTCIPCamera, extending the controllable camera to include NTCIP-specific operations. This class will contain a configuration and status object as necessary to convey NTCIP-specific configuration and status information.

#### **7.4.2.1.35                      NTCIPCameraPositionReference (Class)**

This class represents the NTCIP protocol Camera Position Reference object. This object is used in position commands to configure the speed and direction of movement.

#### **7.4.2.1.36 NTCIPCameraProtocolHdlr (Class)**

This object contains the protocol for communication with a NTCIP Camera.

#### **7.4.2.1.37 NTCIPCameraStatus (Class)**

This structure defines the status data for the NTCIP video camera type.

#### **7.4.2.1.38 PollCameraTask (Class)**

The PollCameraTask is a timer task. When the timer fires it polls a camera by sending a poll command to the camera.

#### **7.4.2.1.39 PushEventSupplier (Class)**

This class provides a utility for application modules that push events on an event channel. The user of this class can pass a reference to the event channel factory to this object. The constructor will create a channel in the factory. The push method is used to push data on the event channel. The push method is able to detect if the event channel or its associated objects have crashed. When this occurs, a flag is set, causing the push method to attempt to reconnect the next time push is called. To avoid a supplier with a heavy supply load from causing reconnect attempts to occur too frequently, a maximum reconnect interval is used. This interval specifies the quickest reconnect interval that can be used. The push method uses this interval and the current time to determine if a reconnect should be attempted, thus reconnects can be throttled independently of a supplier's push rate.

#### **7.4.2.1.40 PutCameraOnlineCmd (Class)**

This class represents the information needed to request a put camera online command to be added on the CommandQueue.

#### **7.4.2.1.41 QueueableCommand (Class)**

A QueueableCommand is an interface used to represent a command that can be placed on a CommandQueue for asynchronous execution. Derived classes implement the execute method to specify the actions taken by the command when it is executed. This interface must be implemented by any device command in order that it may be queued on a CommandQueue. The CommandQueue driver calls the execute method to execute a command in the queue and a call to the interrupted method is made when a CommandQueue is shut down.

#### **7.4.2.1.42 RequestCameraControlCmd (Class)**

This class represents the information needed to request a camera control command to be added on the CommandQueue.

#### **7.4.2.1.43 RequestCameraOverrideCmd (Class)**

This class represents the information needed to request a camera control override command to be added on the CommandQueue.

#### **7.4.2.1.44                      RevokeControlCmd (Class)**

This class represents the information needed to create a revoke camera control command to be added on the CommandQueue.

#### **7.4.2.1.45                      RevokeDisplayCmd (Class)**

This class represents the information needed to create a revoke camera display command to be added on the CommandQueue.

#### **7.4.2.1.46                      ServiceApplication (Class)**

This interface is implemented by objects that can provide the basic services needed by a ChartII service application. These services include providing access to basic CORBA objects that are needed by service applications, such as the ORB, POA, Trader, and Event Service.

#### **7.4.2.1.47                      ServiceApplicationModule (Class)**

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

#### **7.4.2.1.48                      SetSFSBlockedCmd (Class)**

This queueable command is used to block or unblock a camera's stream within a single SFS server.

#### **7.4.2.1.49                      TakeCameraOfflineCmd (Class)**

This class represents the information needed to request a take camera offline command to be added on the CommandQueue.

#### **7.4.2.1.50                      TerminateControlCmd (Class)**

This class represents the information needed to request a terminate camera control command to be added on the CommandQueue.

#### **7.4.2.1.51                      UniquelyIdentifiable (Class)**

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

#### **7.4.2.1.52                      ViconSVFTCameraImpl (Class)**

This class implements the ViconSVFTCamera interface, and inherits from the ControllableCameraImpl class. The ViconSurveyorVFTCameraImpl implements methods of ViconSVFTCamera, extending the controllable camera to include Vicon SVFT-specific operations. This class will contain a configuration and status object as necessary to convey Vicon SVFT-specific configuration and status information.

#### **7.4.2.1.53                      ViconSVFTCameraStatus (Class)**

The ViconSVFTCameraStatus class is used to hold camera status information at the ViconSVFTCamera level. Only ViconSVFTCamera specific information is stored.

#### **7.4.2.1.54                      ViconSVFTPProtocolHdlr (Class)**

This class contains an implementation for Vicon SVFT camera control commands. It translates every camera command (pan, tilt, zoom...) into bytes that a Vicon SVFT camera understands. Then, it uses a CameraControlDevice to send the byte codes to the camera and evaluate responses from the camera.

#### **7.4.2.1.55                      VideoCamera (Class)**

The VideoCamera interface is implemented by objects representing controllable video cameras within the CHART II system. The VideoCamera interface represents a controllable video camera as opposed to the uncontrollable, immovable FixedVideoCamera, the other type of GenericVideoCamera. (The VideoCamera class could have been called the ControllableVideoCamera interface, but since the CHART II video system exists primarily to control controllable video cameras, the camera hierarchy has been arranged to avoid the longish name ControllableVideoCamera.) Current plans call for classes to represent a COHU MPC camera, COHU 3955 camera, Vicon SVFT camera, and NTCIP-compliant camera, and there are interfaces defined for each of these subtypes of VideoCamera. The VideoCamera interface includes the GeoLocatable interface, to someday allow for advanced features such as automatic identification of cameras near traffic events, automatic pointing of cameras to traffic events, etc.

The VideoCamera interface includes all methods common to the three known types of video cameras currently in use by MDSHA, although it is likely to contain a superset of methods which would be implemented by the entire universe of all video cameras which could someday be used. This interface may have to be refined in the event that future brands or models of video cameras might be incorporated under CHART II, but it is an appropriate set of methods for the present day.

#### **7.4.2.1.56                      VideoCameraConfig (Class)**

The VideoCameraConfig structure is used to hold configuration information about VideoCamera objects at the VideoCamera level. Further details about lower-level VideoCamera subclasses are provided by subclasses of VideoCameraConfig.

#### **7.4.2.1.57                      VideoCameraFactory (Class)**

The VideoCameraFactory interface is implemented by factory classes responsible for creating, maintaining, and controlling a collection of VideoCamera objects.

#### **7.4.2.1.58                      VideoCameraImpl (Class)**

The CameraImpl class provides an implementation of the VideoCamera interface, and by extension the VideoSource, SharedResource, CommEnabled, GeoLocatable, and UniquelyIdentifiable interfaces, as specified by the IDL.

This class contains a CommandQueue object that is used to sequentially execute long running operations in a thread separate from the CORBA request threads, thus allowing quick initial responses.

Also contained in this class are VideoCameraConfig and VideoCameraStatus objects (used

to store the configuration and status of the camera), and a VideCameraData object (used to store internal status information which is persisted but not pushed out to clients).

The CameraImpl contains \*Impl methods that map to methods specified in the IDL, including requests to display the camera video on a monitor, remove the camera video from a monitor, put the camera online, put the camera offline, put the camera in maintenance mode (future), or to change (set) the configuration of the camera (future). Some of these requests require (or potentially require) field communications to the device, so each request is stored in a specific subclass of QueueableCommand and added to the CommandQueue. The queueable command objects simply call the appropriate CameraImpl method as the command is executed by the CommandQueue in its thread of execution.

The CameraImpl also contains methods called by the CameraFactory to support the timer tasks of the Camera Service: to look for Cameras with no one logged in at the controlling operations center, and to initiate recovery processing if needed (future).

#### **7.4.2.1.59 VideoCameraStatus (Class)**

The VideoCameraStatus structure is used to hold status information about VideoCamera objects at the VideoCamera level. Further details about lower-level VideoCamera subclasses are provided by subclasses of VideoCameraStatus.

#### **7.4.2.1.60 VideoControlFlashConfig (Class)**

This structure stores configuration information about a flash streaming server configuration that is displaying a camera's image.

#### **7.4.2.1.61 VideoControlFlashStatus (Class)**

This structure contains information about the existence and blocked status of a video source's stream within a Streaming Flash Server (SFS).

#### **7.4.2.1.62 VideoProvider (Class)**

The VideoProvider interface is a generic abstract interface including VideoSource objects (e.g. video cameras) and BridgeCircuit objects. Both VideoSource and BridgeCircuit objects provide video to a VideoCollector, but only VideoSource objects are true origins of video which a typical user would have direct interaction with. BridgeCircuit VideoProvider objects merely pass on video provided from elsewhere in a VideoRoute.

#### **7.4.2.1.63 VideoProviderConfig (Class)**

This structure defines configuration data common to all video sources.

#### **7.4.2.1.64 VideoProviderImpl (Class)**

This class implements the VideoProvider interface as an abstract class. Subclasses for this class are the VideoCameraImpl and BridgeCircuitProviderImpl class.

#### **7.4.2.1.65 VideoProviderStatus (Class)**

The VideoProviderStatus structure is used to hold and transmit status information about VideoProvider objects at the VideoProvider level. Further details about lower-level VideoProvider subclasses are provided by subclasses of VideoProviderStatus.

#### **7.4.2.1.66                      VideoSource (Class)**

The VideoSource interface is implemented by objects which originate video signals, such as video cameras and image generators. Within the user interface, the VideoSource interface represents all video sources which can be put on monitors (i.e., VideoSink objects).

The VideoSource interface includes the SharedResource interface. A VideoSource is controlled by an Operations Center if the VideoSource is in maintenance mode, or if the VideoSource is a camera which has an active control session up.

#### **7.4.2.1.67                      VideoSourceConfig (Class)**

This structure defines configuration data common to all video sources.

#### **7.4.2.1.68                      VideoSourceStatus (Class)**

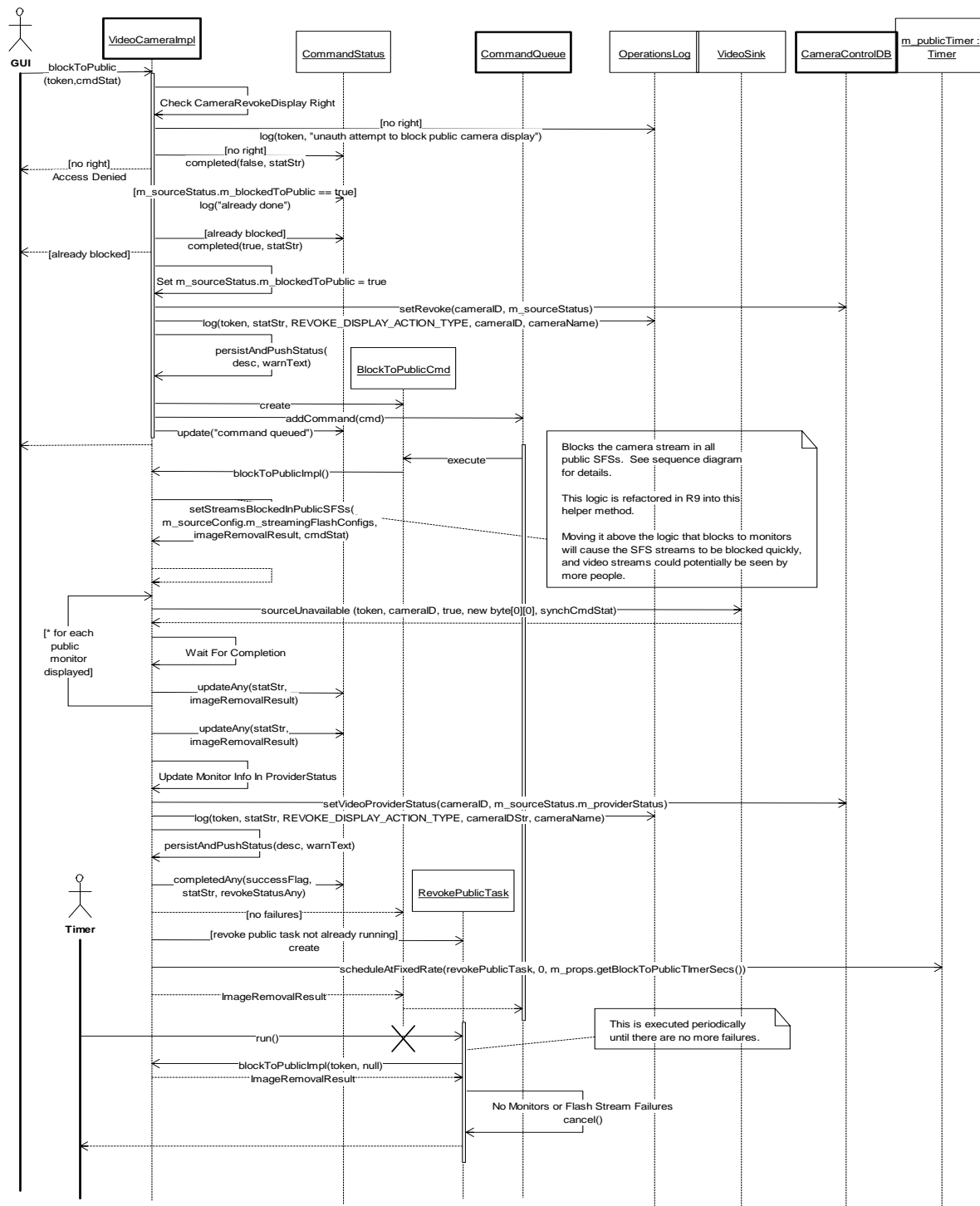
The VideoSourceStatus structure is used to hold and transmit status information about VideoSource objects at the VideoSource level. Further details about lower-level VideoSource subclasses are provided by subclasses of VideoSourceStatus.

## 7.4.3 Sequence Diagrams

### 7.4.3.1 CameraControlModule:BlockToPublic (Sequence Diagram)

This diagram shows the processing when a camera display is blocked to public monitors and Flash streams in SFS servers. (NOTE - This logic is nearly unchanged for R9 the SFS blocking is being changed to check for the new "public" flag and refactored a bit to simplify the code.) After checking rights, the m\_blockedToPublic flag is checked and if it is already set no action is taken to block the camera. If it is not set, the flag is set in memory and persisted to the database, and the status is pushed via a CORBA event. A

BlockToPublicCmd object is created and added to the Command Queue for asynchronous execution. Later, the command is executed and it calls each SFS configured for the camera that is public (see the setStreamsBlockedInPublicSFSs sequence diagram) to block the camera's stream within the SFS. It then calls blockToPublicImpl(), which calls each monitor to display its "No Video Available" source. The status update in memory, persisted, and pushed in a CORBA event. The command status is marked as completed (either successful or failed), but if any failures occurred, a RemovePublicTask object is created and scheduled to execute periodically using a Timer (if such a task is not already scheduled). Sometime later the RevokePublicTask is run, and it calls blockToPublicImpl() again to attempt to block the monitors and public SFS streams. If there are no errors, the RevokePublicTask cancels itself so that it will not be executed again.



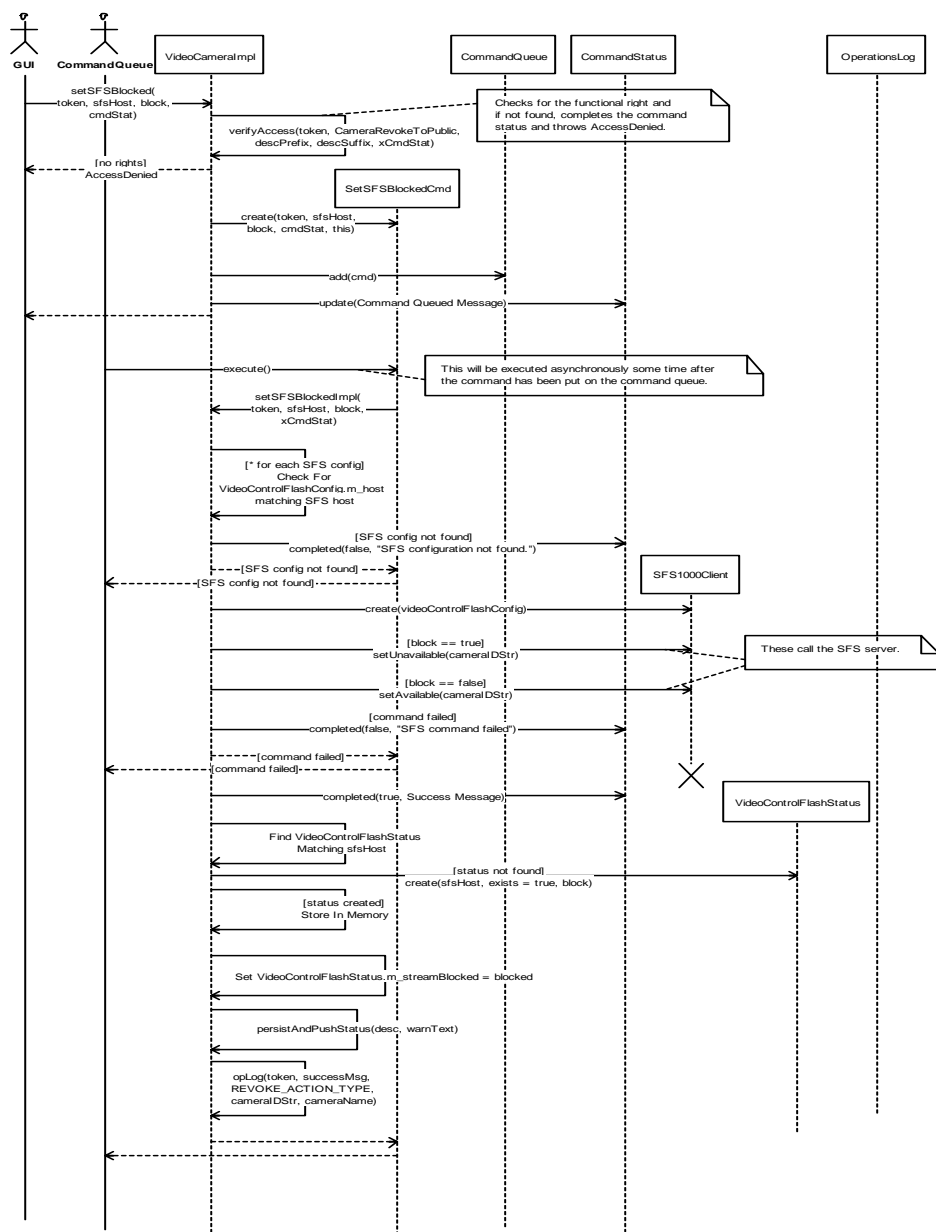
**Figure 7-27. CameraControlModule:BlockToPublic (Sequence Diagram)**



#### 7.4.3.2

#### VideoCameraImpl:setSFSBlocked (Sequence Diagram)

This shows the processing to block or unblock a camera's stream within a single Streaming Flash Server (SFS). The GUI calls `setSFSBlocked()`, and the user's token is checked for the "camera revoke to public" right, and an `AccessDenied` exception is thrown if the user does not have that right. A `SetSFSBlockedCmd` object is created and added to the `CommandQueue` for asynchronous execution. When the command is executed, it calls the `VideoCamera.setSFSBlockedImpl()` method to perform the command. The camera's SFS configurations are searched for a record matching the given `sfsHost` parameter. If no matching SFS configuration is found, the command status is completed with a failure and no further processing is done. If it was found, a new `SFS1000Client` object is created to communicate with the SFS. If the "block" parameter is true, `setUnavailable()` is called; otherwise, `setAvailable()` is called. If the command failed, the `CommandStatus` is completed with a failure and no more processing is done. If successful, the command status is marked as completed. An attempt is made to find a `VideoControlFlashStatus` object matching the `sfsHost` parameter. If none exists, a new status object is created and added to the camera status in memory. The `m_streamBlocked` field is set or cleared, and a call is made to `persistAndPushStatus()` to persist the status to the database and push a new CORBA event. Finally an entry is added to the operations log to document the change.

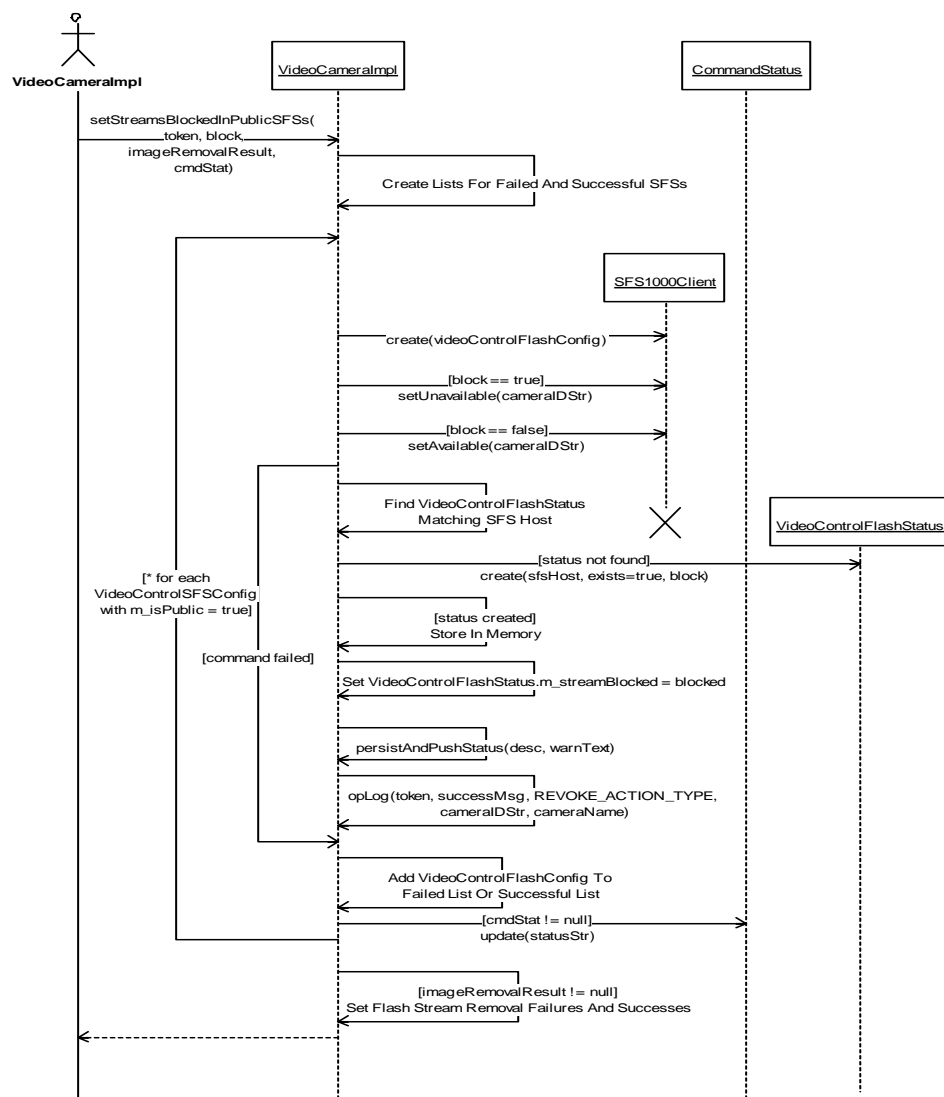


**Figure 7-28. VideoCameraImpl:setSFSBlocked (Sequence Diagram)**

### 7.4.3.3

### VideoCameraImpl:setStreamsBlockedInPublicSFSs (SD)

This diagram shows the processing of a utility method to block or unblock a camera's streams in public SFSs. First, lists are created to store the SFS configurations for successful and failed commands. For each SFS within the camera configuration that has the "public" flag set to true, a SFS1000Client object is created and is used to call setUnavailable() to block the stream or setAvailable() to unblock it. These calls contact the SFS software. If the command was successful, the m\_streamBlocked status is set in memory (creating a new VideoControlFlashStatus if necessary), and the camera status is persisted and pushed and an operations log entry is made. The SFS configuration is added to the failed list or successful list, the command status is updated (if it was not null), and the image removal result is updated (if not null) to store the successful and failed SFS configurations.

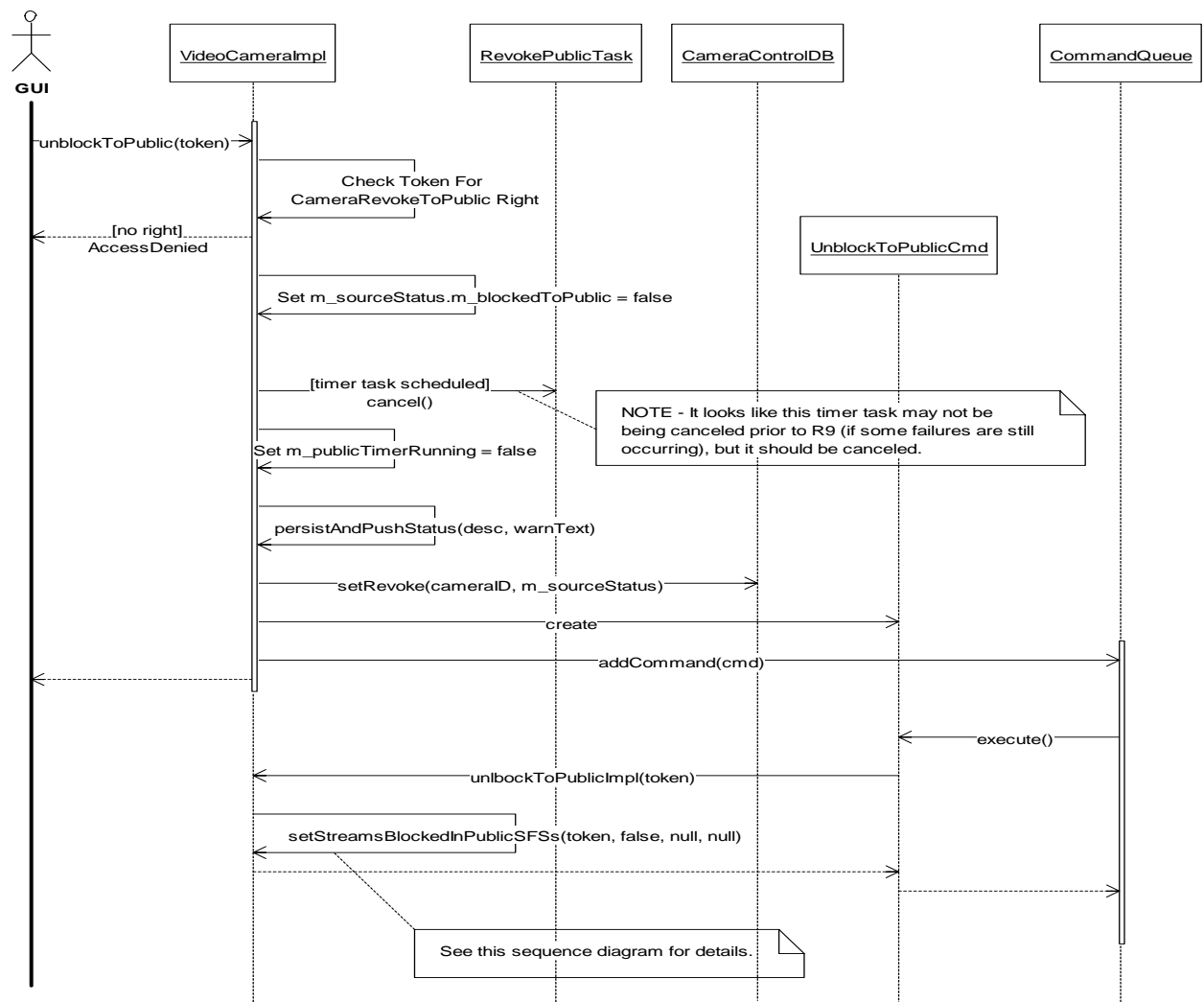


**Figure 7-29. VideoCameraImpl:setStreamsBlockedInPublicSFSs (Sequence Diagram)**

### 7.4.3.4

### VideoCameraImpl:unblockToPublic (Sequence Diagram)

This diagram shows the processing to unblock a camera to public monitors and SFS streams. After checking that the user has sufficient rights, the `m_blockedToPublic` flag is set to false in memory and the `RevokePublicTask` is canceled if it was running. The camera status is persisted and pushed to clients via a CORBA event. An `UnblockToPublicCmd` object is created and added to the command queue, and then sometime later the command is executed and the `setStreamsBlockedInPublicSFSs()` method is called to unblock the camera's stream in each SFS that is configured as public. See that sequence diagram for details.



**Figure 7-30. VideoCameraImpl:unblockToPublic (Sequence Diagram)**

### 7.5.1.1 MiscDataClasses (Class Diagram)

[illegible]

### Figure 7-31. MiscDataClasses (Class Diagram)

This is a base class for push consumers. Derived classes must implement `handleEventData()`.

#### **7.5.1.1.2 DynListSubject (Class)**

This interface is implemented by classes that wish to be capable of being displayed in a dynamic list.

#### **7.5.1.1.3 FolderEnabled (Class)**

This interface provides access to information about an object that can be stored in a folder.

#### **7.5.1.1.4 NotificationShortcutListItem (Class)**

This class represents an item in a notification shortcut list.

#### **7.5.1.1.5 Searchable (Class)**

This interface allows objects to be searched for via a substring search.

#### **7.5.1.1.6 SystemProfileNotificationProperties (Class)**

This class contains functionality for accessing notification settings in the system profile.

#### **7.5.1.1.7 SystemProfileProperties (Class)**

This class is used to cache the system profile properties and provide access to them. It is also used to interact with the server to change system profile settings.

#### **7.5.1.1.8 TempObjectStore (Class)**

This class provides a self cleaning storage area for temporary objects.

#### **7.5.1.1.9 WebAdministered (Class)**

This interface allows the implementing class to be administered via the trader console pages.

#### **7.5.1.1.10 WebDevice (Class)**

This interface contains common functionality for CHART devices.

#### **7.5.1.1.11 WebHARMessageNotifier (Class)**

This interface provides access to HAR notification capabilities for a device (DMS or SHAZAM) that is used to notify the public of a HAR message being broadcast.

#### **7.5.1.1.12 WebOpCenter (Class)**

This class is used to wrap an OperationsCenter object to allow it to be cached in the CHART GUI servlet and to allow the cached data to be accessed within Velocity templates.

#### **7.5.1.1.13 WebSharedResource (Class)**

This interface is implemented by any GUI-side wrapper objects representing CHART shared resources in the system, corresponding to the SharedResource IDL interface.

#### **7.5.1.1.14 WebSharedResourceType (Class)**

This java enum defines the types of shared resources that exist in the system. In addition to the enumeration value, this enumeration contains a description of each shared resource type that can be used for display.

#### **7.5.1.1.15                      WebUniquelyIdentifiable (Class)**

This interface provides functionality for GUI objects that represent UniquelyIdentifiable objects as defined in the IDL.

#### **7.5.1.1.16                      WebVideoSession (Class)**

This class represents a desktop video session, which is an instance of a Flash stream being used by the user.

## 7.5.2 Sequence Diagrams

### 7.5.2.1 ResourceMgmtPushConsumer:handleVideoSessionEnded (Sequence Diagram)

This diagram shows the processing when a "video session ended" CORBA event is handled. The video session ID is used to retrieve the WebVideoSession object from the TempObjectStore. It will only be found if the video session belongs to a user logged into this instance of the GUI servlet. If found, setVideoSessionEnded() is called to mark the video session as ended, storing the reason in the session for later use. (The next time the updateVideoSession request is processed, the WebVideoSession will be removed from the TempObjectStore and the reason will be extracted and passed to the user - see that sequence diagram for details.) The WebVideoSession is also removed from the WebOpCenter's cache, so that the video session will not be displayed in the Video Sessions List, etc..

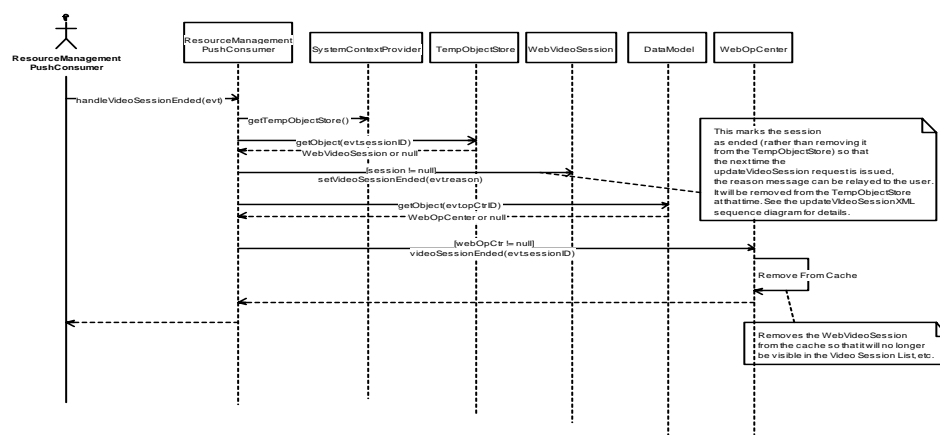


Figure 7-32. ResourceMgmtPushConsumer:handleVideoSessionEnded (Sequence Diagram)



### 7.5.2.2

### WebVideoSession:getWebVideoSessions (Sequence Diagram)

This diagram shows a utility method to get the list of all known video sessions (local to this GUI instance, as well as non-local), optionally specifying the ID of a subject (i.e., video source or tour) to filter the results by subject ID. A HashSet is created to store the sessions. The WebOpCenter objects are retrieved from the DataModel cache. Each WebOpCenter is called to get the WebVideoSessions cached within it. The subject ID and client instance ID are queried from WebVideoSession. If the subject ID was not specified or if the subject ID matches, the WebVideoSession is added to the set if the client instance ID does NOT match (in other words, only those video sessions NOT owned by this GUI instance are added). Next, the local WebVideoSession objects are retrieved from the TempObjectStore (these are the ones owned by this GUI instance). These WebVideoSessions are added to the set if the subject ID was not specified, or if their subject ID matches the specified ID.

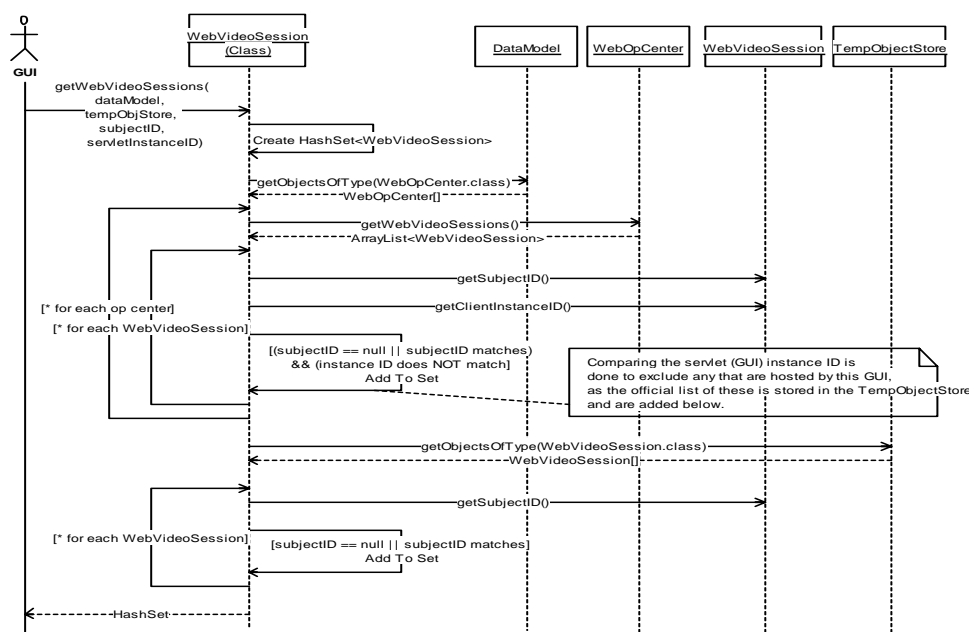


Figure 7-33. WebVideoSession:getWebVideoSessions (Sequence Diagram)

## 7.6 chartlite.data.video-data

### 7.6.1 Class Diagrams

#### 7.6.1.1 GUIVideoDataClasses (Class Diagram)

This diagram shows GUI data classes related to video management.

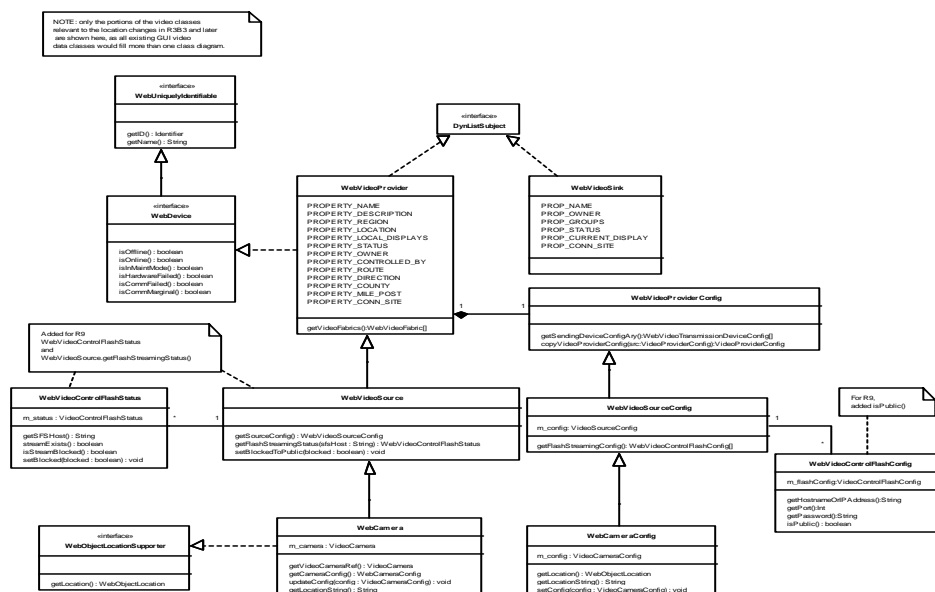


Figure 7-34. GUIVideoDataClasses (Class Diagram)

##### 7.6.1.1.1 DynListSubject (Class)

This interface is implemented by classes that wish to be capable of being displayed in a dynamic list.

##### 7.6.1.1.2 WebCamera (Class)

This class is a wrapper for a VideoCamera CORBA object, used to cache data in the GUI object cache and provide access to the VideoCamera configuration and status data on web pages.

##### 7.6.1.1.3 WebCameraConfig (Class)

This class wraps the VideoCameraConfig structure defined in the IDL and provides accessor methods.

##### 7.6.1.1.4 WebDevice (Class)

This interface contains common functionality for CHART devices.

#### **7.6.1.1.5                      WebObjectLocationSupporter (Class)**

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

#### **7.6.1.1.6                      WebUniquelyIdentifiable (Class)**

This interface provides functionality for GUI objects that represent UniquelyIdentifiable objects as defined in the IDL.

#### **7.6.1.1.7                      WebVideoControlFlashConfig (Class)**

This class wraps a VideoControlFlashConfig struct for display on a web page.

#### **7.6.1.1.8                      WebVideoControlFlashStatus (Class)**

This class wraps contains information about whether a video stream exists within an SFS server, and whether it is currently blocked. Note that in R9, this is the last status known to CHART, and does not take into account any externally issued blocking commands, as there is no way of querying the SFS server via its API for stream existence or blocked status.

#### **7.6.1.1.9                      WebVideoProvider (Class)**

This class wraps the VideoProvider CORBA reference and stores cached configuration and status for fast local access.

#### **7.6.1.1.10                    WebVideoProviderConfig (Class)**

This class wraps the VideoProviderConfig structure defined in the IDL and provides accessor methods.

#### **7.6.1.1.11                   WebVideoSink (Class)**

This class wraps the VideoSink CORBA reference and stores cached configuration and status for fast local access.

#### **7.6.1.1.12                   WebVideoSource (Class)**

This class wraps the VideoSource CORBA reference and stores cached configuration and status for fast local access.

#### **7.6.1.1.13                   WebVideoSourceConfig (Class)**

This class wraps the VideoSourceConfig structure defined in the IDL and provides accessor methods.



#### **7.7.1.1.4 MainServlet (Class)**

This class is the main class of the servlet. It handles all requests and dispatches them to the appropriate request handler. It also acts as a RequestHandlerSupporter, which is passed to each request handler to help them process requests.

#### **7.7.1.1.5 MaintenancePortalContentMapping (Class)**

This class is a PortalContentMapping for the device maintenance portal. It has a hashtable that provides a mapping of standard GUI content templates to custom versions for use when the user is logged into the device maintenance portal. Note that not all CHART pages are customized for the maintenance portal - when no customized page exists, this mapping returns the original page.

#### **7.7.1.1.6 NavLinkRights (Class)**

This class provides user rights checking for the servlet. It contains a user's token and provides easy to use methods that can check the presence of functional rights, combinations of rights, or even rights that are specific to the object the user wishes to use.

#### **7.7.1.1.7 org.apache.velocity. VelocityServlet (Class)**

The base class for the Velocity template engine. This template engine is used to provide dynamic content from the CHART GUI Servlet. The web pages are code in templates using velocity specific macros. The code in the servlet loads data that will be shown on the page into a velocity Context, and this VelocityServlet class is used to merge the content with the template to create HTML for the browser to display.

#### **7.7.1.1.8 PortalContentMapping (Class)**

This interface specifies a method to be implemented by classes that provide a mapping of standard CHART web pages to versions that are customized for a portal which provides a different view of the system. It also specifies a method that allows portals to specify content that is always to be displayed in a popup, even if the portal prefers to not use popups. This feature exists for special content such as pages that allow the user to listen to audio via a browser plugin.

#### **7.7.1.1.9 PortalType (Class)**

This enumeration defines the portal type the user has logged into. FullView represents the full versioned GUI. DeviceMaintenance represents the device maintenance portal, which is tailored to use by device maintenance personnel.

#### **7.7.1.1.10 RequestAction (Class)**

This class contains information about an action that can be invoked via a request handler. The action parameter is specified in the URL as the "action" parameter, or as the last part of the servlet path. The user logged out policy specifies what the servlet should do if this action is requested when the user is logged out.

#### **7.7.1.1.11 RequestHandler (Class)**

This interface specifies methods that are to be implemented by classes that are used to

process requests.

**7.7.1.1.12 RequestHandlerMapping (Class)**

This class provides a mapping between an action and the request handler used to process a request for that action.

**7.7.1.1.13 RequestHandlerSupporter (Class)**

This interface is implemented by any class that can provide access to objects or methods that are helpful to request handlers.

**7.7.1.1.14 ServletDB (Class)**

This class is used by the CHART GUI servlet to access CHART GUI specific data that is stored in the database.

**7.7.1.1.15 ServletProperties (Class)**

This class provides access to properties defined in the chart gui's properties file.

**7.7.1.1.16 UserLoggedOutPolicy (Class)**

This enumeration specifies the types of actions that may be specified for responding to a request that is received when the user is logged out.

**7.7.1.1.17 UserLoginSessionImpl (Class)**

This class is the implementation of the CORBA UserLoginSession interface. It will be served from the GUI and will be passed to the OperationsCenter on login. It will also store the access token returned from the OperationsCenter.



#### **7.8.1.1.3 DynListReqHdlrDelegate (Class)**

This class helps request handlers support dynamic lists. Requests to view, sort, or filter dynamic lists can be passed from a request handler to this class, provided the URL used for the requests contain parameters required by this class, such as the id of the list, the property name, and/or the filter value.

#### **7.8.1.1.4 EditObjectLocationSupporter (Class)**

This interface provides functionality allowing the location data to be edited. (For example, the target of the edited location may be an existing object, or it may be a form data object for creating a new object).

#### **7.8.1.1.5 EditVideoCameraLocationSupporter (Class)**

This class is used to support editing the location of an existing or new VideoCamera.

#### **7.8.1.1.6 FormUtil (Class)**

This class contains methods for handling form processing.

#### **7.8.1.1.7 MonitorListSupporter (Class)**

This class is a DynListDelegateSupporter that provides Monitor specific functionality to the generic DynListReqHdlrDelegate.

#### **7.8.1.1.8 SelectMonitorsList (Class)**

This class represents the select monitors dynamic list

#### **7.8.1.1.9 SelectMonitorsListSupporter (Class)**

This class provides functionality required by the DynListReqHdlrDelegate object for the Select Monitors page.

#### **7.8.1.1.10 SelectVideoSourcesList (Class)**

This class is a DynList used to select video sources. It supports single or multiple select models. The caller specifies a target action that should be invoked when the selection is complete and a caller ID that should be passed back to the caller. After selection is complete, a request url of the following format will be created.

#### **7.8.1.1.11 VideoSessionListSupporter (Class)**

This class provides functionality allowing the Video Session dynamic list to be displayed.

#### **7.8.1.1.12 VideoSinkReqHdlr (Class)**

This class is a request handler that processes requests related to video sinks such as Monitors.

#### **7.8.1.1.13 VideoSourceConfigReqHdlr (Class)**

This class handles requests related to video source configuration.

#### **7.8.1.1.14 VideoSourceListSupporter (Class)**

This class is a DynListDelegateSupporter that provides Video Source specific functionality



to the generic DynListReqHdlrDelegate.

**7.8.1.1.15                      VideoSourceSelectListSupporter (Class)**

This class provides functionality required by the DynListReqHdlrDelegate object for the Video Source Selection List page.

**7.8.1.1.16                      VideoTourReqHdlr (Class)**

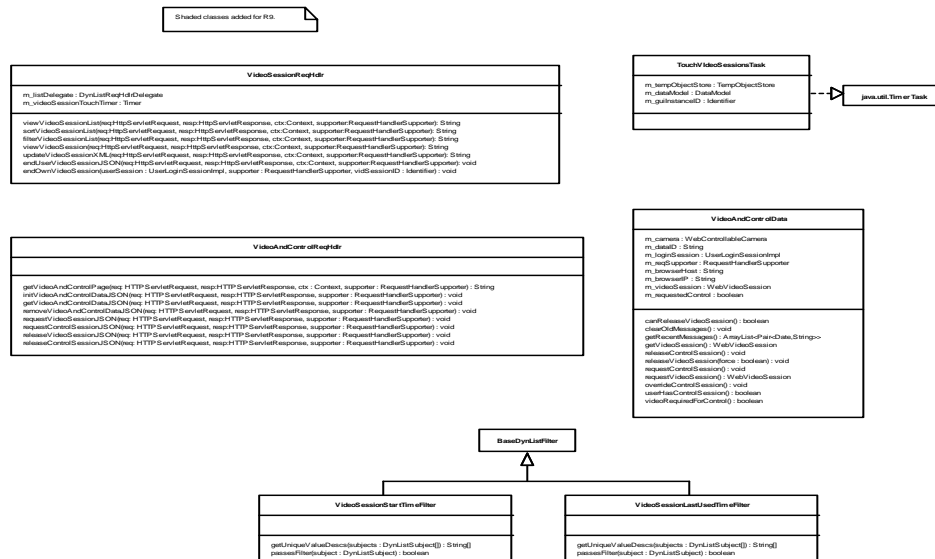
This class handles requests for Video Tour functionality.

**7.8.1.1.17                      ViewVideoSourceReqHdlr (Class)**

This class handles requests that allow the user to view a video source or list of video sources.

## GUIVideoServletClasses2 (Class Diagram)

This diagram shows additional GUI classes involved in processing video-related requests.



**Figure 7-37. GUIVideoServletClasses2 (Class Diagram)**

#### 7.8.1.2.1 BaseDynListFilter (Class)

This abstract class provides a base implementation of the DynListFilter interface.

### 7.8.1.2.2 java.util.TimerTask (Class)

This class is an abstract base class which can be scheduled with a timer to be executed one or more times.

### 7.8.1.2.3 TouchVideoSessionsTask (Class)

This class is a timer task that is called periodically to inform CHART of which video sessions are still active.

#### 7.8.1.2.4 VideoAndControlData (Class)

This class represents data for a combined video / camera control dialog.

#### 7.8.1.2.5 VideoAndControlReqHdlr (Class)

This class handles requests related to managing the sessions for the combined Video and Camera Control dialog.

#### 7.8.1.2.6 VideoSessionLastUsedTimeFilter (Class)

This class will be used to filter the Video Session List by last used time

#### **7.8.1.2.7                      VideoSessionReqHdlr (Class)**

This class handles requests related to video sessions.

#### **7.8.1.2.8                      VideoSessionStartTimeFilter (Class)**

This class will be used to filter the Video Session List by start time.

## 7.8.2 Sequence Diagrams

### 7.8.2.1 ControlVideoSourceReqHdlr:processSetSFSBlocked (Sequence Diagram)

This diagram shows the processing for blocking or unblocking a video source to an SFS. The videoSourceID parameter is used to retrieve the WebVideoSource wrapper object from the GUI cache. After checking the user's rights, the sfsHost parameter is parsed to specify which SFS to block/unblock the stream for, and the "block" parameter specifying whether to block or unblock the stream is also parsed. A command status is created, and the VideoSource object is called to block or unblock the display to the SFS. The browser is redirected to view the Command Status page.

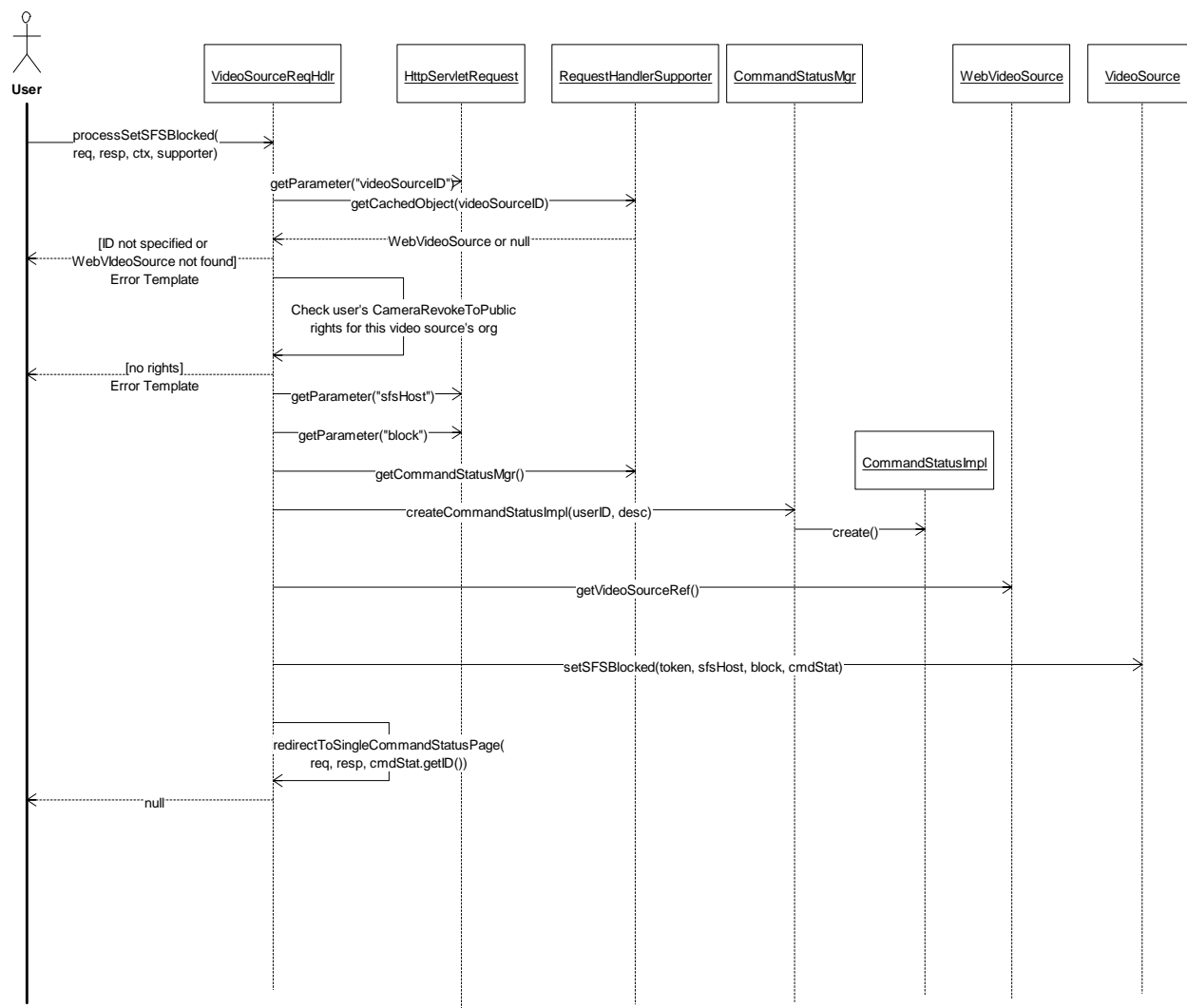
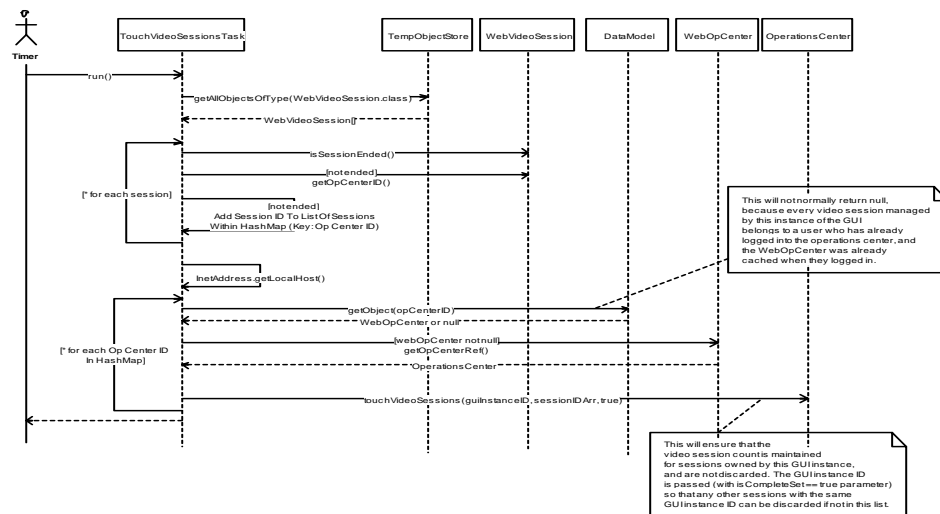


Figure 7-38. ControlVideoSourceReqHdlr:processSetSFSBlocked (Sequence Diagram)

### 7.8.2.2

### TouchVideoSessionsTask:run (Sequence Diagram)

This diagram shows the processing to periodically inform CHART services of which video sessions are being managed by this GUI. The TouchVideoSessionsTask's run() method is called periodically on a timer. It gets all WebVideoSession objects that are stored in the TempObjectStore, as this is the set of all video sessions for users logged into this GUI instance. The session IDs from active video sessions are put into a HashMap keyed on operations center ID. Each OperationsCenter responsible for active sessions is called with the list of video session IDs that this GUI instance owns. The sessions are passed with the GUI's instance ID so that the CHART server can remove any sessions it has for this GUI that are not in the new list. The server will update the last used time for the video sessions and will push a CORBA event so that other GUIs can update their caches as well.

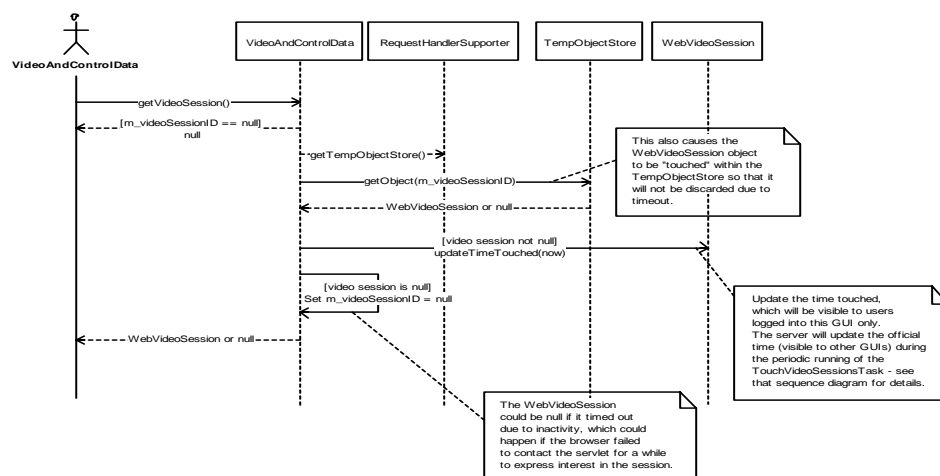


**Figure 7-39. TouchVideoSessionsTask:run (Sequence Diagram)**

### 7.8.2.3

### VideoAndControlData:getVideoSession (Sequence Diagram)

This diagram shows a utility method to get an existing video session managed by the VideoAndControlData object. If the m\_videoSessionID is not null, the TempObjectStore is called to retrieve the WebVideoSession object and "touch" the object so it will not be discarded by the TempObjectStore for a while. If returned, the time touched is updated within the WebVideoSession object so that it can be displayed to users logged into the local GUI (the time seen by other GUI instances will be updated when the TouchVideoSessionsTask runs - see that sequence diagram for details). If for some reason the WebVideoSession was NOT found in the TempObjectStore, the session ID is cleared.



**Figure 7-40. VideoAndControlData:getVideoSession (Sequence Diagram)**

## 7.8.2.4

### VideoAndControlData:releaseVideoSession (Sequence Diagram)

This diagram shows the processing to release a video session from within the Video / Control dialog. The `releaseVideoSession()` method is called, passing a "force" parameter, which will be true if the user is closing the window, or false if attempting to close the video portion without releasing control of the camera. First `getVideoSession()` is called, which gets the current video session or returns null if it does not exist. If it does not exist, there is nothing to do so it returns. If it does exist, a check is made to see whether the user has a control session for the camera, and if so whether the current video session is needed for local display of the camera. If the video session is needed, it returns without releasing the session. To release the video session, the video session object is removed from the `TempObjectStore` and from the `WebOpCenter` cache. Then the `OperationsCenter` is called to inform the server that the session has ended, so that it can decrease its counter of allocated sessions. (Note that even if this call fails, the server will clean up the session after it times out because the GUI servlet will no longer be reporting the session as "in use"). The `m_videoSessionID` is set to null, indicating that no session exists, and a success message is added to be returned in the next status update.

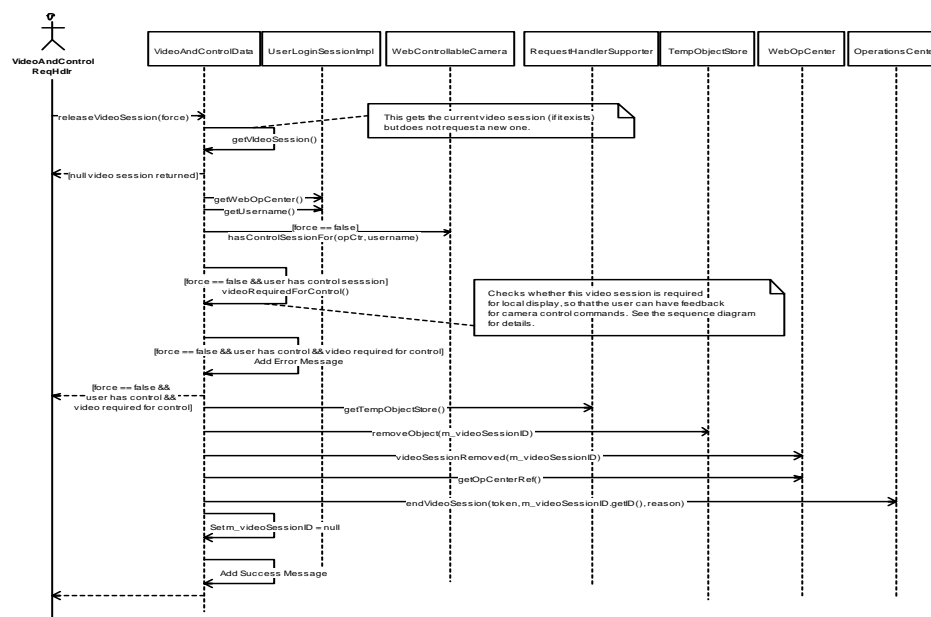


Figure 7-41. VideoAndControlData:releaseVideoSession (Sequence Diagram)

## 7.8.2.5 VideoAndControlData:requestControlSession (Sequence Diagram)

This diagram shows a utility method to request (or override) control of a camera within the Video And Control Dialog. The `m_requestedControl` flag is set so that the browser will know to display the control portion of the window. The `WebControllableCamera` is queried to see if there is a control session for any user, and if this user has the control session. If there is a control session for another user and an override was not requested, an error message will be appended to the list of messages (which will be included in the next status update) before returning. Next, `videoRequiredForControl()` is called to check whether the camera is displayed on a local monitor or in another of the user's desktop video sessions (see that sequence diagram for details). If the camera is not displayed locally or in another of the user's sessions, `requestVideoSession()` is called to request a new session (see that sequence diagram for details). If a video session already exists for this window it will be used; otherwise, a new video session will be requested. If a video session is required but cannot be obtained, an error message will be appended and the method will return. Otherwise, a request is made to the `ControllableVideoCamera` to request (or override) control. This is an asynchronous command, and the command status and camera status will indicate when the control session is acquired.

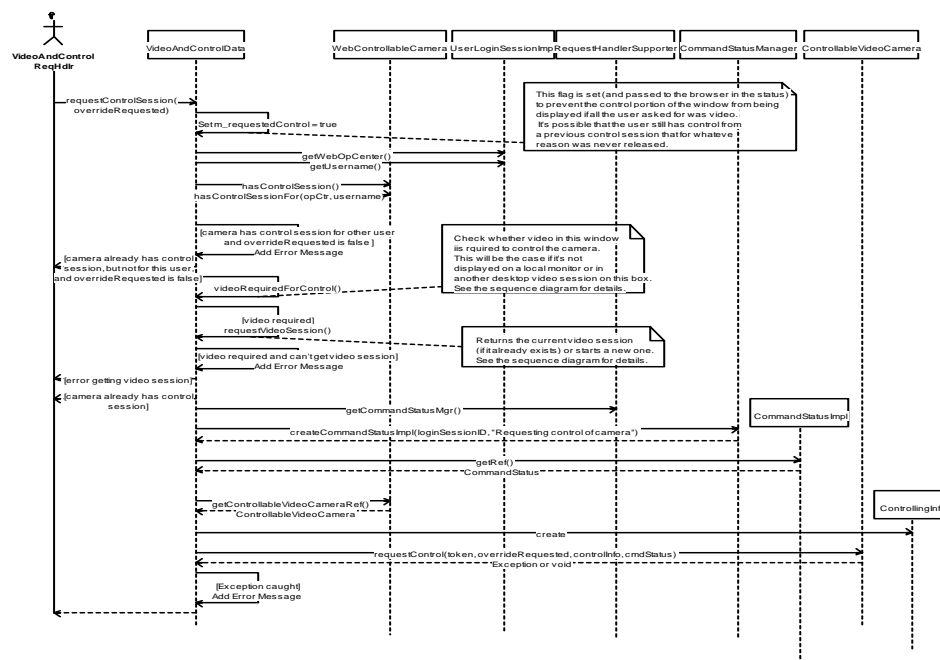


Figure 7-42. VideoAndControlData:requestControlSession (Sequence Diagram)



## 7.8.2.6

### VideoAndControlData:requestVideoSession (Sequence Diagram)

This diagram shows the processing to request a video session from within a combined Video / Camera Control dialog. First it calls the `getVideoSession()` method to see if a video session already exists in the `TempObjectStore`, and returns it if it does exist. If it does not exist, the user's `OperationsCenter` is called to request the video session. If the request is denied due to the resource limit being reached, an error message is added to the list of messages stored in the `VideoAndControlData`. If successful, a new `WebVideoSession` is created and added to the `TempObjectStore()`, the `m_videoSessionID` is set for later use, and a success message is added to the `VideoAndControlData` object before returning the new `WebVideoSession`.

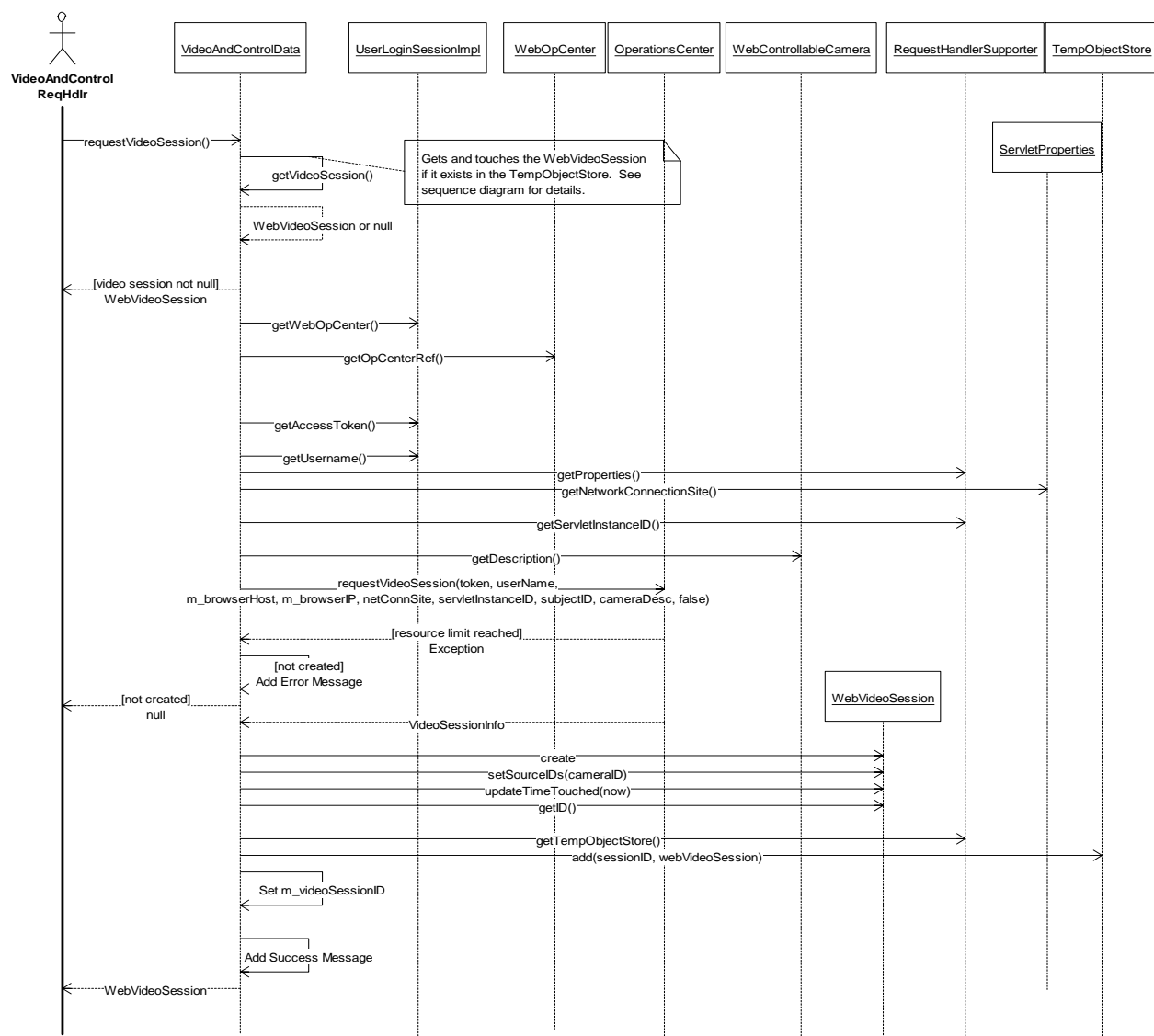


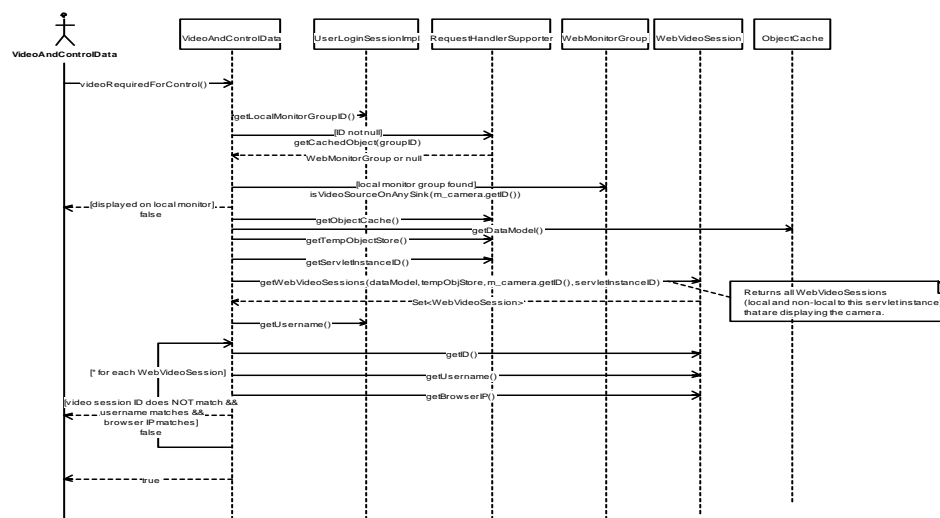
Figure 7-43. VideoAndControlData:requestVideoSession (Sequence Diagram)

### 7.8.2.7 Diagram)

### VideoAndControlData:videoRequiredForControl

(Sequence

This diagram shows a utility method to determine whether this window's video session (which may or may not exist) is necessary to be displayed in order for the camera to be controlled, if or when it is controlled, to ensure that the user can see the effects of camera control commands. It gets the user's local monitor group (if it is specified) and calls `isVideoSourceOnAnySink()` to see if it is displayed on a local monitor, and returns false if it is. Otherwise, it calls `getWebVideoSessions()` (see that sequence diagram) to get all known `WebVideoSessions` (local to this servlet instance, as well as non-local ones) that are displaying this camera. For each `WebVideoSession`, it gets the ID, user name, and browser IP of the session. A session with a matching ID is ignored, as we are testing for the existence of \*other\* video sessions displayed on the user's desktop. If the user name and browser IP matches the user's name / IP it returns false, indicating that this window's video session is not required for control; otherwise, it returns true.



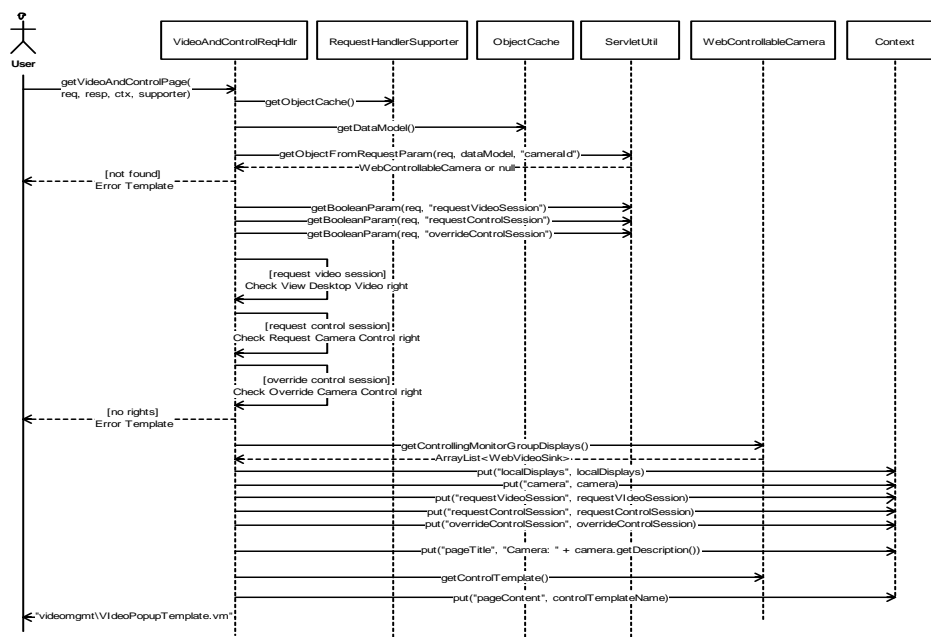
**Figure 7-44. VideoAndControlData:videoRequiredForControl (Sequence Diagram)**

## 7.8.2.8 Diagram)

## VideoAndControlReqHdlr:getVideoAndControlPage

(Sequence

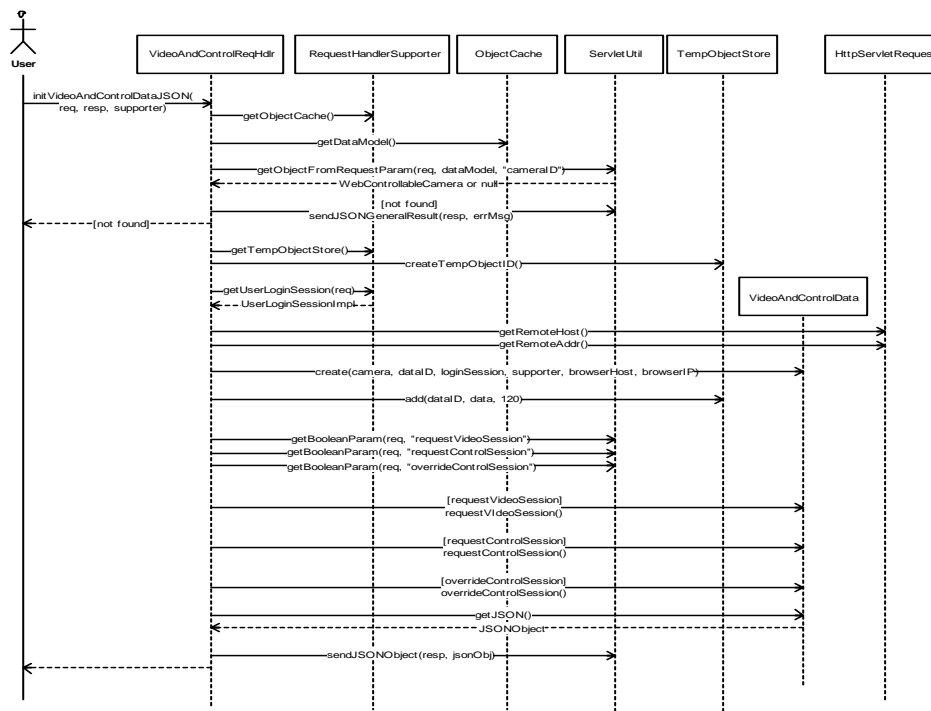
This diagram shows the processing to display the combined Video and/or Control dialog for a controllable camera. The camera ID parameter is used to retrieve the WebControllableCamera object from the cache. The initialization flags for requesting a video session, requesting a control session, or overriding a control session are also parsed from parameters and the user's rights are checked. The Velocity context is populated, and the camera's control template is put into the context, and the video popup template path is returned.



**Figure 7-45. VideoAndControlReqHdlr:getVideoAndControlPage (Sequence Diagram)**

### 7.8.2.9 VideoAndControlReqHdlr:initVideoAndControlDataJSON (Sequence Diagram)

This diagram shows the processing to initialize the data object representing the state of the combined Video and/or Control dialog for a controllable camera. The camera ID parameter is used to retrieve the WebControllableCamera object from the cache. A VideoAndControlData object is created and added to the TempObjectStore cache to ensure that the object will be cleaned up if the window is closed unexpectedly. Request parameters are parsed to request a video session, request camera control, and/or override camera control. If these flags are specified, the VideoAndControlData object is called to perform the appropriate action (see the corresponding sequence diagram for details). The information in the VideoAndControlData object is sent back to the browser via a JSON response so that the page can be updated to display / store the information.

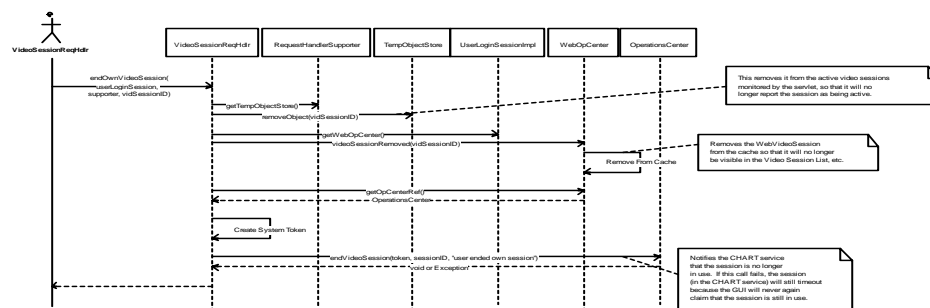


**Figure 7-46. VideoAndControlReqHdlr:initVideoAndControlDataJSON (Sequence Diagram)**

### 7.8.2.10

### VideoSessionReqHdlr:endOwnVideoSession (Sequence Diagram)

This diagram shows the processing when a video session is ended for a user's own session (which typically happens implicitly when a video window is closed). The VideoSessionReqHdlr removes the WebVideoSession object from the TempObjectStore, so that it will no longer be monitored by the GUI. It also removes the WebVideoSession object from the WebOpCenter corresponding to the user's operations center, so that it will no longer be displayed in the lists. It also calls the CHART service to remove the session to release the session resource immediately and notify the other GUIs. If for some reason this is NOT called (for example, if the browser crashes or the window fails to detect that it is being closed) the WebVideoSession will be removed automatically from the TempObjectStore after some time, as no "update session" requests will continue to "touch" the object to prevent it from being discarded by the TempObjectStore.



**Figure 7-47. VideoSessionReqHdlr:endOwnVideoSession (Sequence Diagram)**

## 7.8.2.11

### VideoSessionReqHdlr:endUserVideoSession (Sequence Diagram)

This diagram shows the processing when a user's video session is ended by an administrator. After checking the user rights, the video session ID request parameter is used to attempt to look up the WebVideoSession object from the TempObjectStore. It will only exist in the TempObjectStore if the video session is local to this GUI servlet instance. If found, setVideoSessionEnded() is called to mark the session as ended and to pass a reason string to the user. (The session will be removed from the TempObjectStore later when the next updateVideoSession() request is processed - see that diagram for details). The WebOpCenter is then retrieved from the cache and the CHART service is called to end the video session (which releases the resource). If successful, or if there was a local video session, the WebVideoSession is also removed from the WebOpCenter's cache of video sessions. For non-local sessions, additional processing will be done when the CORBA event is handled (see the handleVideoSessionEnded sequence diagram for details).

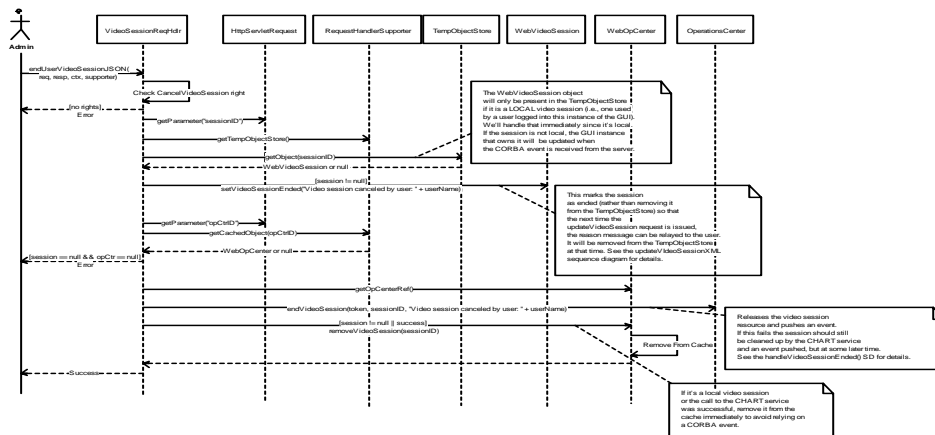


Figure 7-48. VideoSessionReqHdlr:endUserVideoSession (Sequence Diagram)

### VideoSessionReqHdlr:updateVideoSessionXML (Sequence Diagram)

[illegible]

### Figure 7-49. VideoSessionReqHdlr:updateVideoSessionXML (Sequence Diagram)

### 7.8.2.13

### VideoSessionReqHdlr:viewVideoSession (Sequence Diagram)

This diagram shows the processing when the user views a tour or video source (not a controllable camera, as those will be handled separately). After checking the user rights, the VideoSessionReqHdlr gets the tourID and providerID (one of which should be specified) and looks up the WebVideoTour or WebVideoProvider from the servlet's cache. If the requested object is not found an error is displayed. Next a video session is requested. If resource limit has been reached (i.e., the operations center has too many open video sessions) an exception will be thrown by the server and an error will be displayed to the user; otherwise, the VideoSessionInfo will be returned, and a WebVideoSession wrapper is created and added to the TempObjectStore. The video session and the tour (or video provider) wrapper objects are placed into the Velocity context, and the video template is returned.

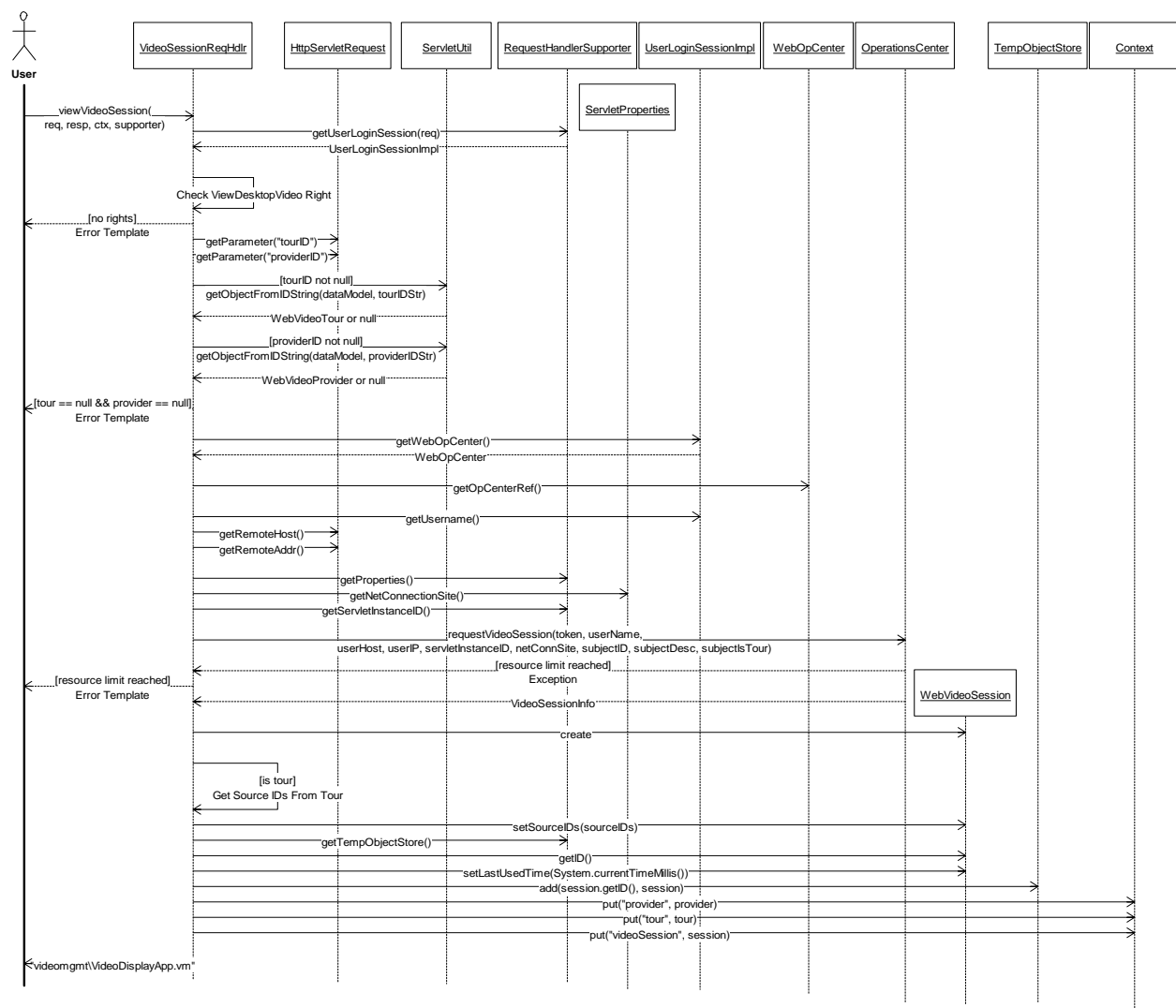


Figure 7-50. VideoSessionReqHdlr:viewVideoSession (Sequence Diagram)



#### 7.8.2.14 VideoSessionReqHdlr:viewVideoSessionList (Sequence Diagram)

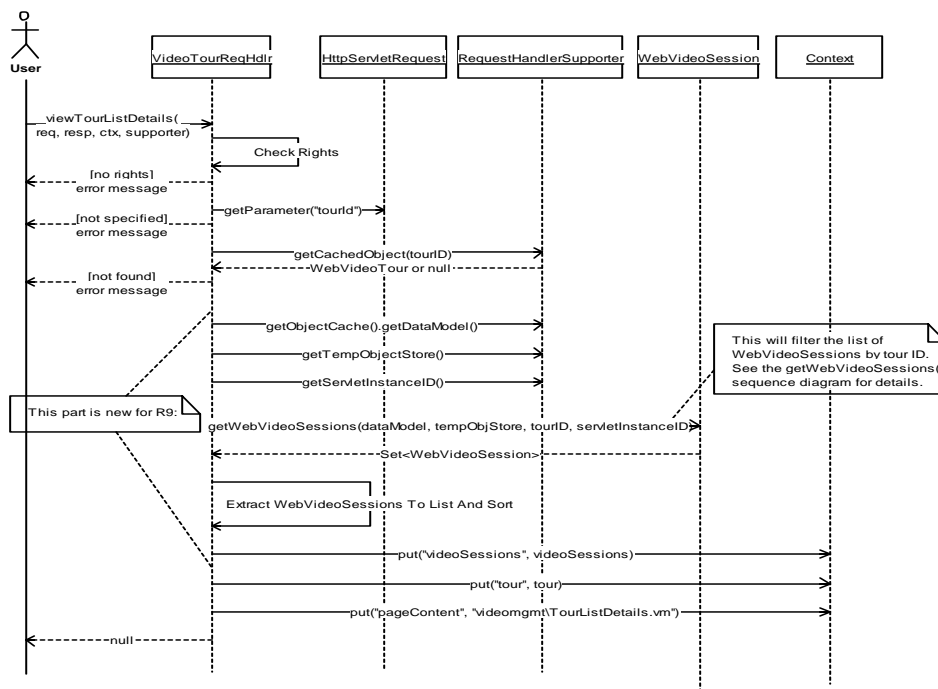
This diagram shows how the Video Session list is displayed. The user's request causes `viewVideoSessionList()` to be called. This calls the `DynListReqHdlrDelegate` to process the `viewDynList()`. It tries to get the `DynList` object from the `TempObject Store` (it will be there only if previously displayed). If not found, it calls the `VideoSessionListSupporter` to create the `DynList` object. The columns, filters, and sort comparators are created, and then the `DefaultDynList` object is created with the supported columns. The initial filter values are set to display the user's video sessions by default, and the `DynList` object is returned. It is then added to the `TempObjectStore` and the initial column visibility is set up (applying any previously-saved settings). The response is then sent causing the browser to be redirected, so that the ID of the `DynList` will be used in the URL. (After being redirected, it will enter `viewVideoSessionList()` again, but it will find the `DynList` in the `TempObjectStore` and will not create a new one.) If the `DynList` was retrieved from the `TempObjectStore` (i.e., was not created), a call is made to the `VideoSessionListSupporter` to get the `DynList` subjects (i.e., the `WebVideoSession` objects to display). This calls the static `WebVideoSession.getWebVideoSessions()` utility method to get all of the video sessions (see the `getWebVideoSessions()` sequence diagram for details). The subjects are then returned and set into the `DynList`, at which point the current sort is applied (if applicable) to sort the subjects. The Velocity context is populated with the necessary objects, and the Velocity template name is returned for rendering.



### 7.8.2.15

### VideoTourReqHdr:viewTourListDetails (Sequence Diagram)

This diagram shows the processing to display the Video Tour Details page. After checking user rights, the "tourId" parameter is used to retrieve the WebVideoTour object from the cache. Then the static WebVideoSession.getWebVideoSessions() utility method is called, passing the tour ID to return a set of matching video sessions. (See the getWebVideoSessions() sequence diagram for details). The returned video session objects are put into a list and sorted, and the tour and sessions are put into the Velocity context for rendering.



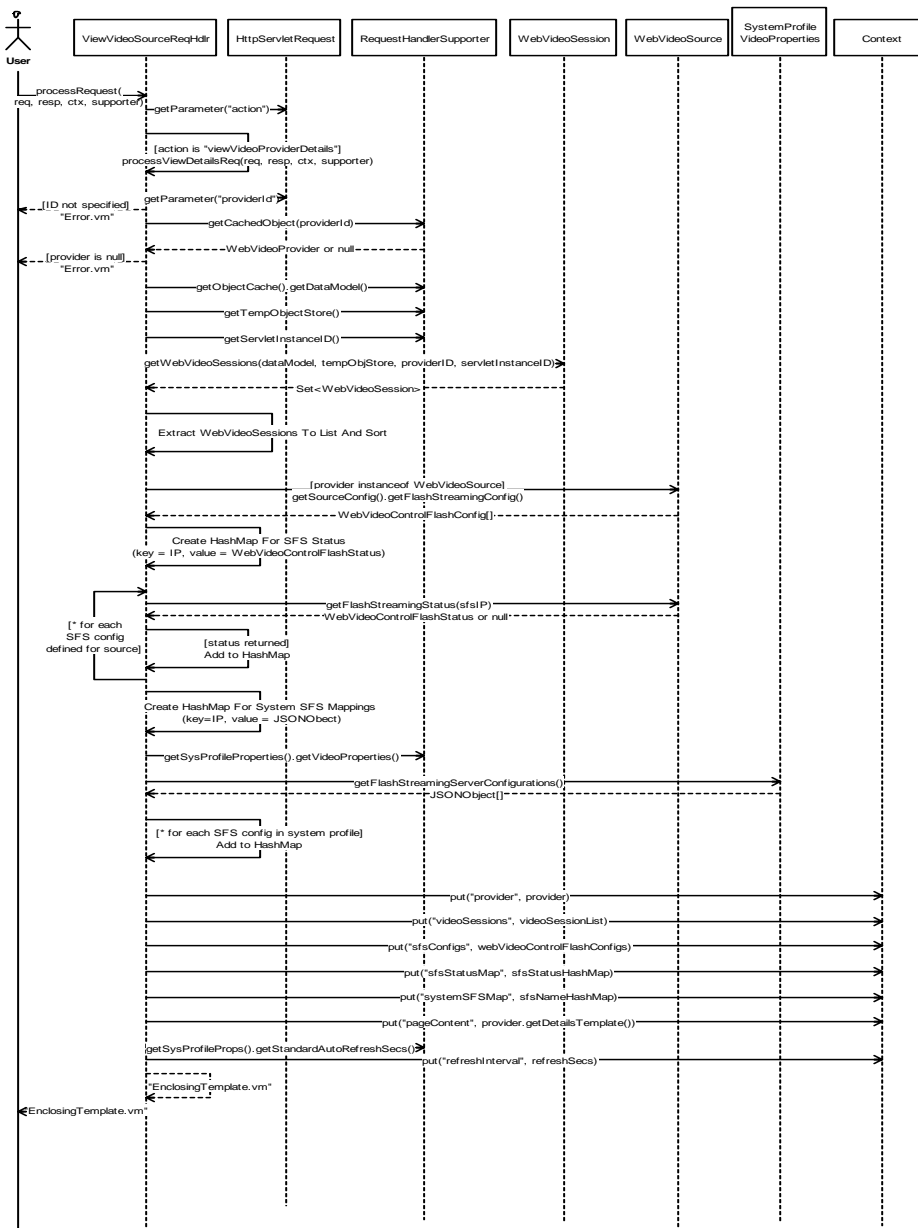
**Figure 7-52. VideoTourReqHdr:viewTourListDetails (Sequence Diagram)**

#### 7.8.2.16 Diagram)

#### ViewVideoSourceReqHdlr:processViewDetailsReq

(Sequence

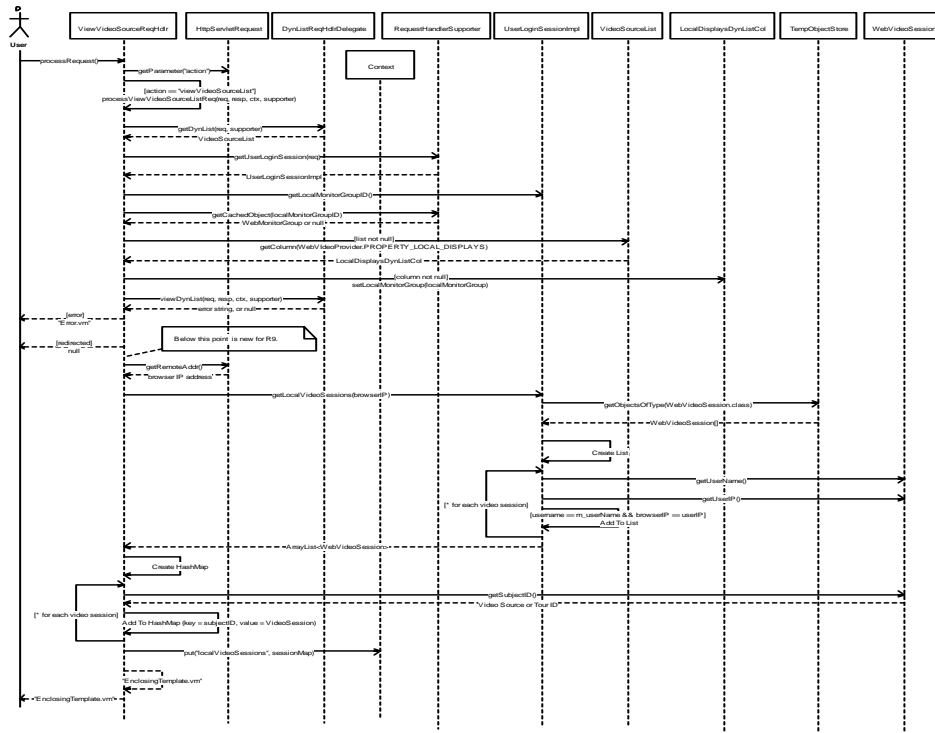
This diagram shows the processing when the Video Source Details page is displayed. The "providerId" parameter is used to retrieve the WebVideoProvider object from the servlet's cache. For R9, the static WebVideoSession.getWebVideoSessions() utility method is called, passing the provider ID to return a set of matching video sessions. (See the getWebVideoSessions() sequence diagram for details). The returned video session objects are put into a list and sorted. Also for R9, the Streaming Flash Server (SFS) status and configuration will be shown for each SFS server configured for the video source. To display this, first the SFS configuration is retrieved from the WebVideoSource's configuration. Then a HashMap is made for easy lookup of the SFS status, keyed on the SFS server IP (or host...it is assumed to match the config) by retrieving the WebVideoControlFlashStatus objects from the WebVideoSource and putting them in a HashMap. For display of the SFS mnemonic (name) corresponding to the IP/host and other info, the system SFS configurations are retrieved from the System Profile, and are put into another HashMap keyed on SFS IP. The provider, the SFS configurations for the source, the SFS status map, and the SFS map are all put into the Velocity context so they can be displayed on the web page. (The SFS status map and/or SFS map may not contain entries for each configuration, but that will be handled gracefully by the Velocity template logic.)



**Figure 7-53. ViewVideoSourceReqHdlr:processViewDetailsReq (Sequence Diagram)**

## ViewVideoSourceReqHdlr:processViewVideoSourceListReq

This diagram shows the processing when the Video Source List is displayed. The dynamic list object (which was previously created and put in the temporary object store) is retrieved, and the column corresponding to the "Local Displays" column is retrieved and the local monitor group object is retrieved and is set into the column object so that the Velocity template can display the local monitors. The DynListReqHdlrDelegate is then called (via viewDynList()) to prepare the Velocity context for displaying the list (it may also create a new dynamic list, if necessary, and then redirect the response to view the new list). New for R9, the local video sessions are queried from the UserLoginSessionImpl, using the browser IP address obtained from the request. This gets the cached WebVideoSession objects from the TempObjectStore, and returns the subset matching the user's name and browser IP. The sessions returned are then put into a lookup table (keyed by the video source or tour ID) so that they can be easily looked up for each video source or tour, and displayed on the Video Source List page.



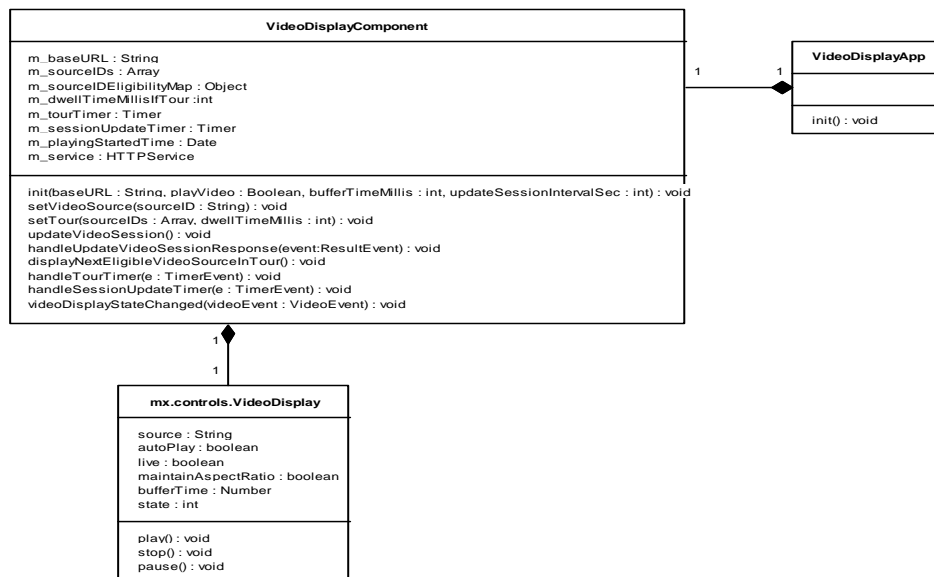
**Figure 7-54. ViewVideoSourceReqHdlr:processViewVideoSourceListReq (Sequence Diagram)**

## 7.9 video-flex

### 7.9.1 Class Diagrams

#### 7.9.1.1 video\_flex\_classes (Class Diagram)

This diagram shows video-related Flex classes



**Figure 7-55. video\_flex\_classes (Class Diagram)**

##### 7.9.1.1.1 mx.controls.VideoDisplay (Class)

This class is an Adobe Flex control for displaying video. Only the subset of its functionality used by CHART is shown here.

##### 7.9.1.1.2 VideoDisplayApp (Class)

This class is a simple Flex application to display a single video component.

##### 7.9.1.1.3 VideoDisplayComponent (Class)

This class wraps a VideoDisplay component and adds CHART application functionality.

## 7.9.2 Sequence Diagrams

### 7.9.2.1 VideoDisplayApp:init (Sequence Diagram)

This diagram shows the processing when the Video Display Flex application is initialized. The "creation complete" event causes Flex to call the VideoDisplayApp's init() method. The application parameters are parsed, and these parameters are passed to the VideoDisplayComponent. Button event listeners are added, as is an event listener for video display state changes. The autoplay and buffer time attributes are set, the current state is set to normal (which causes the non-fullscreen controls to be displayed) and the size combo box is populated and selected. Then, an initial call to updateVideoSession() is made, which issues a request to the servlet. The servlet will return a response that includes the video source eligibility status, and when the response is received the video can begin to buffer (see the handleUpdateSessionResponse SD). A timer is created to periodically issue the update session request. If it is a tour, a tour timer is created to manage the video source switching within the tour. When the timer fires, handleTourTimer() is called (see that sequence diagram for details).

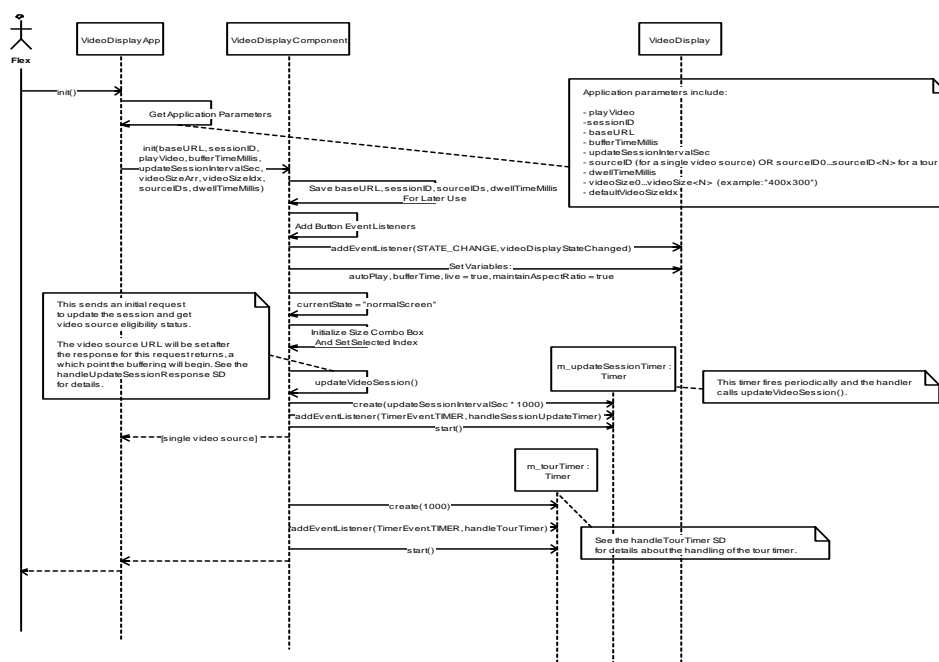
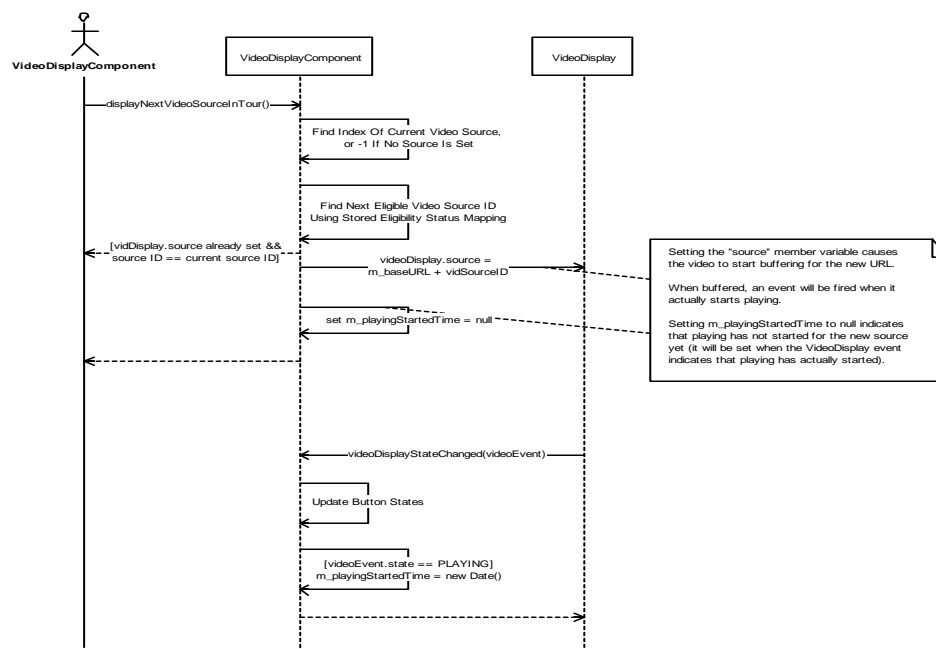


Figure 7-56. VideoDisplayApp:init (Sequence Diagram)



### 7.9.2.2 VideoDisplayComponent:displayNextEligibleVideoSourceInTour (Sequence Diagram)

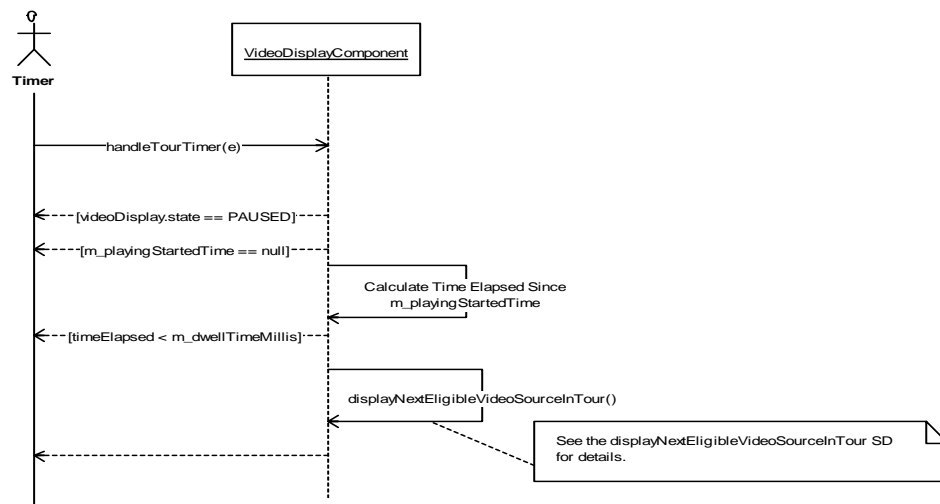
This diagram shows the processing to display the next video source in a tour. Starting with the current video source (or starting at the beginning of the list, if no source is already set), it finds the next eligible video source in the tour using the stored eligibility status flags (returned in the last session update response). If there is only one video source that is eligible (i.e., the next eligible source is the same as the current one) it returns and does not change the source. Otherwise, it sets the VideoDisplay's "source" member to a new URL corresponding to the new video source, which causes the VideoDisplay Flex component to start buffering the video. The `m_playingStartedTime` is cleared (set to null). Later, when the video has been buffered and begins playing, the VideoDisplay control triggers an event which is handled. The buttons are updated to reflect the current playing state, and if the video is playing the `m_playingStartedTime` is set to the current time, so that the tour timer can ensure that the current video source gets a proper dwell time.



**Figure 7-57. VideoDisplayComponent:displayNextEligibleVideoSourceInTour (Sequence Diagram)**

### 7.9.2.3 VideoDisplayComponent:handleTourTimer (Sequence Diagram)

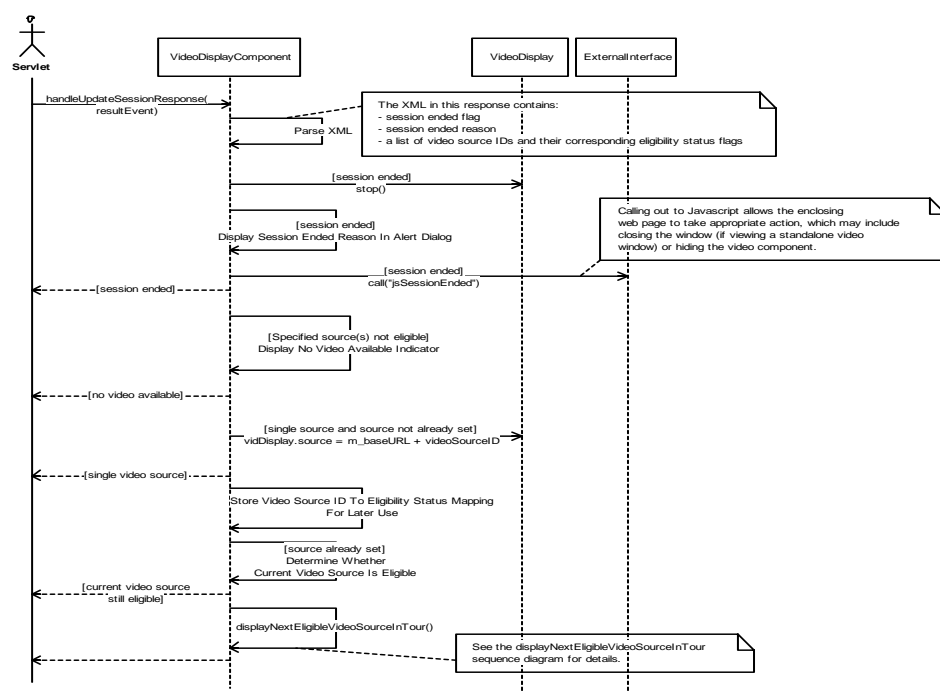
This diagram shows the processing when the tour timer is triggered. If the video display state is "paused" it simply returns to avoid switching the video source. Also if `m_playingStartTime` is null, the current source (if any) has not started playing yet, so again it just returns (taking no action). Otherwise, the time since `m_playingStartTime` is calculated. If more time has elapsed than the dwell time, `displayNextEligibleVideoSourceInTour()` is called to display the next source (see that sequence diagram for details).



**Figure 7-58. VideoDisplayComponent:handleTourTimer (Sequence Diagram)**

### 7.9.2.4 VideoDisplayComponent:handleUpdateSessionResponse (Sequence Diagram)

This diagram shows the processing when the VideoDisplayComponent receives a response to an "update session" request. The response can include a "session ended" flag (and reason string), as well as a list of video source IDs and eligibility flags for those sources (it will be a single source unless it's a tour). After parsing the XML and examining the "session ended" flag, if the session has ended the VideoDisplay Flex component is called to stop playing, an alert dialog is displayed, a call is made to the Javascript in the enclosing web page. If the specified video source(s) is/are all ineligible, a No Video Available indication is displayed. If the session is a single source, it sets the VideoDisplay.source member (if not already set) to initialize the video (i.e., to begin buffering the video) and then returns. If it's a tour, it determines whether the currently playing source is still eligible. If the source is no longer eligible, a call is made to display the next eligible source (see the displayNextEligibleVideoSourceInTour sequence diagram for details).

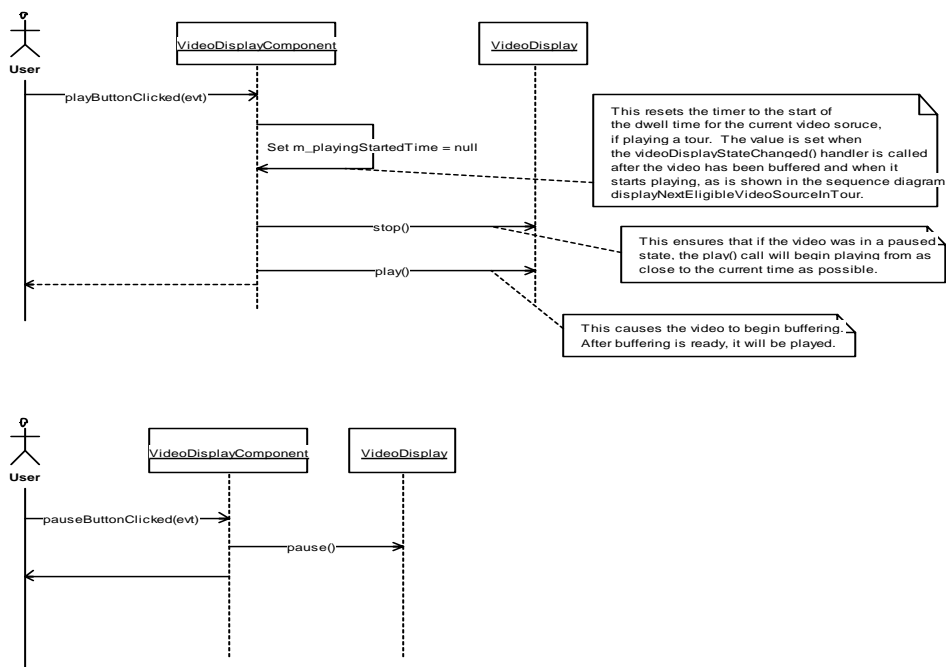


**Figure 7-59. VideoDisplayComponent:handleUpdateSessionResponse (Sequence Diagram)**

### 7.9.2.5

### VideoDisplayComponent:playAndPause (Sequence Diagram)

This diagram shows the processing for two operations: Play (or "Live") and Pause. When the Play/Live button is clicked, the playing started time is set to null so that the current video source will be displayed again for the full dwell time, if it is a tour. The VideoDisplay control's stop() method is then called, to ensure that if video was previously paused, it will not resume from that point - it will start playing at close to the current time. The play() method is then called, which starts the buffering process. When buffering is complete, Flex will trigger "display state changed" event which is handled by videoDisplayStateChanged(), at which time the playing started time will be set to start counting time for the current video source. In the second part of the diagram: if the user clicks the Pause button, the VideoDisplayComponent simply calls the VideoDisplay's pause() method which pauses the video.



**Figure 7-60. VideoDisplayComponent:playAndPause (Sequence Diagram)**

## 8 Use Cases – G4 RTMS

The use case diagrams depict new functionality for the CHART TSS features. The use case diagrams exist in the Tau design tool in the Release9 area. The sections below indicate the title of the use case diagrams that apply to the TSS.

### 8.1 ConfigureTSS (Use Case Diagram)

This Use Case Diagram identifies the actions that can be performed to configure the TSS. Starting in R9, two models of TSS are supported, “X3 RTMS” (previously called simply “RTMS”) and “G4 RTMS”, so starting in R9, users can specify and change the model type of a detector. Additionally up to twelve zones can be configured for the newer G4 RTMS.

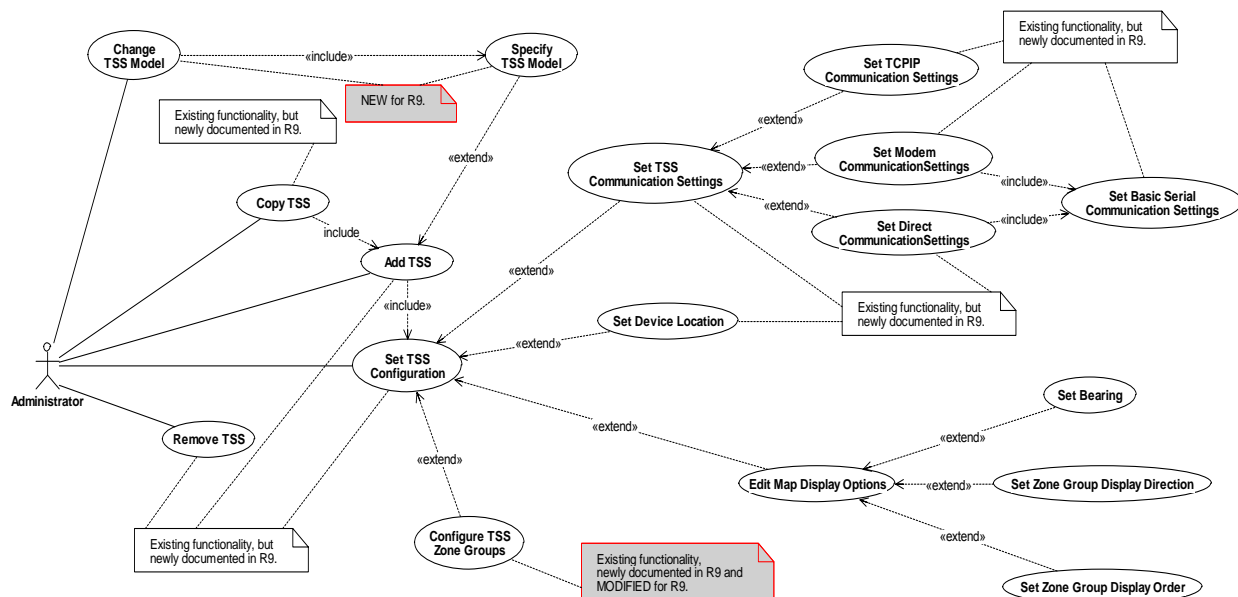


Figure 8-1. ConfigureTSS (Use Case Diagram)

#### 8.1.1 Add TSS (Use Case)

A user with appropriate privileges may add a TSS to the system. Certain settings for the TSS must be specified when the TSS is added, such as model type and communication settings. Others are optional. When copying an existing TSS, existing values are pulled in from the source TSS, so the "required" data is already specified; editing them becomes optional.

### **8.1.2 Change TSS Model (Use Case)**

An administrator can change the model of a TSS if the TSS is offline. The user can select X3 RTMS or G4 RTMS. When a change occurs, any data which cannot be used by the target TSS model (such as zones 9-12 of a G4 RTMS being changed to an X3 RTMS) will be lost.

### **8.1.3 Configure TSS Zone Groups (Use Case)**

An administrator can configure zone groups for a TSS. Each zone group can contain one or more zones (lanes) moving in the same direction of travel. Generally, a TSS has two zone groups - one for all the lanes in one direction and one for all the lanes in the other direction. (Other examples are one zone group in one direction, two zone groups in each direction, and two zone groups in one direction only.) Starting in R9, different models of TSS can support different numbers of zones (X3 RTMS supports 8 zones, G4 RTMS supports 12 zones).

### **8.1.4 Copy TSS (Use Case)**

A user with appropriate privileges may add a TSS to the system by copying an existing TSS. All data is copied from the source TSS, and any data can be edited or not at the discretion of the operator before submitting the request to add the TSS.

### **8.1.5 Edit Map Display Options (Use Case)**

An administrator can edit the map display properties for any TSS that has a defined location (lat/lon). The map display options can be edited with the TSS in any mode (online, offline or maintenance mode). The map display properties include: TSS bearing, zone group display direction, and zone group display order.

### **8.1.6 Remove TSS (Use Case)**

An administrator can remove an offline TSS from the CHART system.

### **8.1.7 Set Basic Serial Communication Settings (Use Case)**

An administrator can set basic communication settings for a device communicating over a modem or serial port. These include baud rate, data bits, stop bits, parity, and flow control.

### **8.1.8 Set Bearing (Use Case)**

An administrator can specify the bearing for any TSS that has a defined location (lat/lon). A bearing of 0 degrees shall mean the bearing is due east. The bearing shall grow counter-clockwise such that a bearing of 90 indicates due north, 180 indicates due west, and 270 indicates due south. A TSS will not have a defined bearing when it is initially created. The TSS bearing is used to orient the zone groups for the TSS on the maps.

### **8.1.9 Set Device Location (Use Case)**

An administrator can set the location information for a device, including: location description, latitude/longitude, state, county, route type, route number (or name, and flag to indicate which to use), direction (including bidirectional directions), proximity to intersecting feature (intersection, state or county milepost, or exit number), and intersecting feature information. The intersecting feature information will include state milepost number or intersecting route type, route number (or name, and flag to indicate which to use). The system will validate the geographical coordinates (if entered) against system-wide bounds defined in the system profile to make sure that they are not unreasonable. The location description is a required field, and it will be automatically generated by the system but may be overridden by the user. The system will prompt for confirmation when the user attempts to override the generated location description.

### **8.1.10 Set Direct CommunicationSettings (Use Case)**

An administrator can set direct port serial communication settings for a device which supports direct port communications. These include basic serial communication settings, and a list of port managers together with a named port to be used on each port manager.

### **8.1.11 Set Modem CommunicationSettings (Use Case)**

An administrator can set modem communication settings for a device which supports communication via modem. These include default phone number, basic serial communication settings, modem type (ISDN or POTS), and an ordered list of port managers (FMSs) together with a phone number to be used by each port manager to connect to the modem.

### **8.1.12 Set TCPIP Communication Settings (Use Case)**

An administrator can set TCP/IP communication settings to be set for devices that support TCP/IP communications. The TCP/IP communication settings include the IP address of the device and the port where the device listens for connections.

### **8.1.13 Set TSS Communication Settings (Use Case)**

An administrator can set communication settings for a TSS. These will be either TCP/IP communications settings, modem communications settings, or direct communication settings.

### **8.1.14 Set TSS Configuration (Use Case)**

An administrator can set the configuration of a TSS. This can be done as part of adding a TSS or modifying an existing TSS. Some basic TSS settings are part of this use case (such as polling interval, device logging flag, maintaining and owning organizations, etc.), and other settings are captured within extended (optional) use cases (some of which can be

shared by other device configuration use cases). Indicating whether this TSS should run its Built-inTest at the system wide TSS BIT time is part of this use case as of R9.

#### **8.1.15 Set Zone Group Display Direction (Use Case)**

An administrator can specify the display direction for each zone group. The display direction will indicate how the zone group should be displayed on maps. A user can specify that a zone group should either be displayed on maps or not be displayed on maps. For zone groups that are displayed on maps, a user can specify whether they are displayed using an arrow that points in the direction of the TSS bearing or using an arrow that points in the opposite direction (180 degrees opposed) of the TSS bearing. When a user changes a zone group from not displayable on maps to displayable on maps, the system shall display a warning that the zone group name may be displayed on the Internet map. A zone group will be set to not displayable on maps when it is initially created.

#### **8.1.16 Set Zone Group Display Order (Use Case)**

An administrator can specify the display order for each zone group relative to other zone groups of the TSS with the same display bearing. Starting at the location of the TSS, zone groups with a lower display order will appear first on the map and zone groups with higher display orders will appear further away from the TSS latitude/longitude position.

#### **8.1.17 Specify TSS Model (Use Case)**

An administrator can specify the model of a TSS just being added to the system or of an existing TSS. As of R9, the choices are X3 RTMS and G4 RTMS.



## 8.2 ManageTSS (Use Case Diagram)

This Use Case Diagram describes activities involved in managing or manipulating TSSs. Some actions can be performed by users of various types (such as operators, administrators, and device maintenance personnel), including changing the TSS communication mode, editing TSS-related System Profile parameters, and commanding a Built-in Test (BIT) to be run on a TSS.

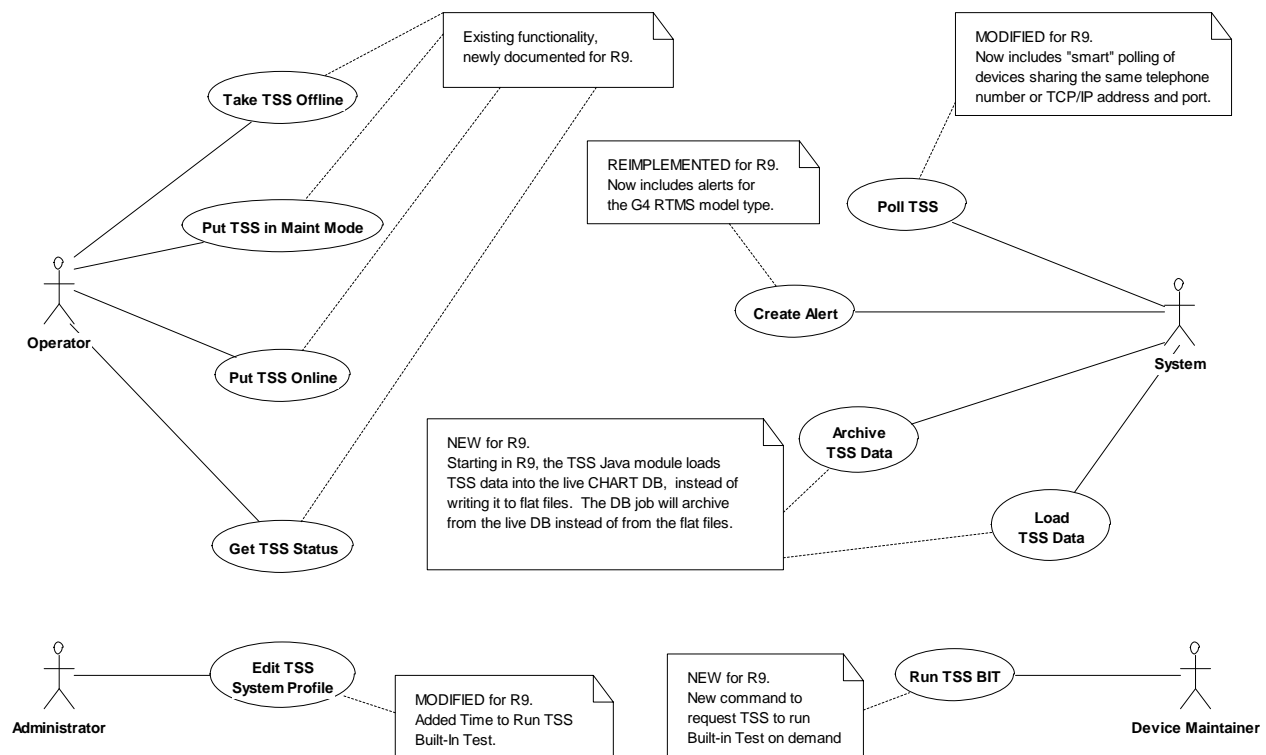


Figure 8-2. ManageTSS (Use Case Diagram)

### 8.2.1 Archive TSS Data (Use Case)

The system will archive each set of traffic data as it is acquired by polling internal detectors. (Traffic data for external detectors is not archived.)

### 8.2.2 Create Alert (Use Case)

The system will create an alert when necessary.

### 8.2.3 Edit TSS System Profile (Use Case)

An administrator can change the System Profile relative to TSSs. This includes TSS speed zone ranges, the TSS speed threshold percentage and enabled flag, and the time of day that

TSS Built-in Test (BIT) is to run on TSSs enabled to run scheduled BIT.

#### **8.2.4 Get TSS Status (Use Case)**

A user with appropriate rights can request a call back to the server to ensure the GUI is displaying the latest TSS status (and configuration).

#### **8.2.5 Load TSS Data (Use Case)**

Upon restart, the system will load the most set of traffic data as it was acquired by polling, for each internal detector, provided that data is not too old. (Traffic data for external detectors is not archived.)

#### **8.2.6 Poll TSS (Use Case)**

The system automatically polls each online TSS at its configured polling rate for traffic data. This traffic data is displayed to users on the TSS List and TSS Details pages and pushed out RITIS and the Intranet/Internet Maps through the Data Exporter. New for R9, the system will manage and smartly poll TSSs that are on the same phone number or same TCP/IP address and port, so that the sharing devices are polled sequentially on the same phone call or TCP connection.

#### **8.2.7 Put TSS in Maint Mode (Use Case)**

A user with appropriate rights can put a TSS in maintenance mode.

#### **8.2.8 Put TSS Online (Use Case)**

A user with appropriate rights can put a TSS online.

#### **8.2.9 Run TSS BIT (Use Case)**

A device maintainer can run Built-in Test (BIT) on a TSS upon demand, if the TSS is in maintenance mode. This test returns results in a command status and updates the hardware failure details (faults detected, if any, and time BIT last run) on the TSS Details Page.

#### **8.2.10 Take TSS Offline (Use Case)**

A user with appropriate rights can take a TSS offline.

## 8.3 ViewTSS (Use Case Diagram)

This Use Case Diagram describes read-only activities involved in viewing TSSs and their data.

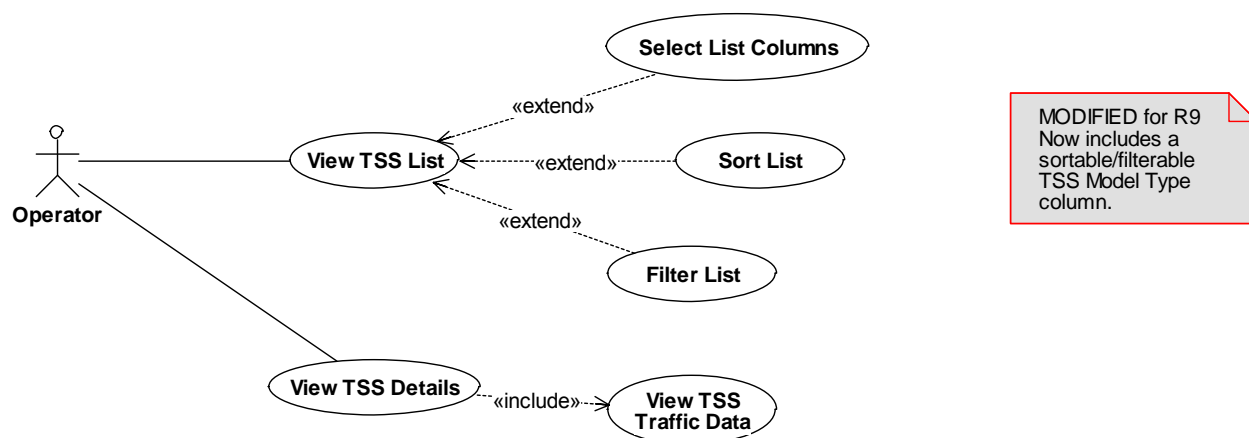


Figure 8-3. ViewTSS (Use Case Diagram)

### 8.3.1 Filter List (Use Case)

The system allows lists to be filtered to include only rows that have specified values in a column. Multiple filters can be used on a single list, and the system allows all filters to be removed to view all items in the list. Columns that have a large range of possible values may not support filtering (for example, columns that display device messages or display travel route or detector speeds), however, in some of these cases, custom hardcoded filters may be available, such as "Blank" and "Not Blank" for device messages, and various speed ranges for speeds).

### 8.3.2 Select List Columns (Use Case)

The system allows the user to choose which columns are to appear in a list and which columns are to be hidden. Certain columns such as the name of an item may not be permitted to be hidden.

### 8.3.3 Sort List (Use Case)

The system allows lists to be sorted using values in a specific column of the list. Ascending and descending sorts are supported. Not all columns support sorting.

### 8.3.4 View TSS Details (Use Case)

A user with appropriate rights can view the details of a TSS device. The details show

include the name, location, owning and maintaining organizations, operations center to receive alerts, polling interval, device logging flag, execute scheduled BIT flag, results of last BIT including time run and faults detected, communication settings, zone group configuration, status, and traffic data. The location includes the location description, county or region, state, route information, intersecting feature data, and latitude/longitude. The route information includes either a free-form route description, or the route type and route. The intersecting feature data includes either an intersecting road, milepost, or exit. The location also includes a link to show the device on the map. The zone group configuration shows for each zone group a one-up number, description, direction of travel, posted speed, display direction (toward or away from device bearing), detection zones comprising each zone group. The communication settings shall indicate if a modem port, direct port, or TCP/IP is used to communicate with the TSS. When a modem port is used, the system shall display the default phone number, serial port parameters, port manager timeout and port managers (with phone numbers from each port manager). When a direct port is used, the system will display the serial port parameters, port manager timeout and port managers (with port name to be used from each port manager). When TCP/IP is used, the system shall display the IP address and port used by the SHAZAM. Device phone numbers, access codes, port manager timeouts, port type, IP addresses and TCP ports are considered sensitive information and will only be shown if the user has the right to view sensitive information for the TSS (based on its owning organization). The serial port parameters include the baud rate, data bits, stop bits, parity, and flow control. TSS Details also includes the traffic data as described in the View TSS Traffic Data Use Case.

### **8.3.5 View TSS List (Use Case)**

The system will allow a user with appropriate rights to view the list of TSSs defined in the system. Data shown for each TSS will include the description (name) and location, average speeds (per zone), status, last update time, route, direction, milepost (State numbering), county, port managers, network connection site, owning organization, maintaining organization, and model type. To save screen space, the visible columns will be selectable. Several columns will be hidden by default to save space. The user will be able to sort the list by any of the columns listed above. The user will be able to filter the list by any of the columns listed above except name, location, last update time, and milepost.

### **8.3.6 View TSS Traffic Data (Use Case)**

A user with appropriate rights can view the more recently retrieved speed volume, and occupancy of each configured zone group. If the user does not have rights to view speed, but does have rights to view speed ranges, a speed range (e.g., 0-30, 30-50, or 50+) will be displayed, but no volume or occupancy percentage. If the user does not have rights to view speed or speed ranges, no data will be shown.

## 9 Detailed Design – G4 RTMS

---

### 9.1 Human-Machine Interface

The human-machine interface for TSS devices is changed in R9 to allow specifying and viewing model types of detectors. The new model type is the “G4 RTMS”. The old model type, known till now as simply “RTMS”, will now be known as the “X3 RTMS”. Both models will support an ability to command them to run Built-in Test (BIT), and to display results, including (new for R9) the time BIT was last executed, in addition to the brief description of faults found which has been provided all along in CHART. The G4 RTMS can support up to 12 zones (lanes) of traffic detection, which CHART will also support, in terms of configuring zone groups and displaying detected traffic data (speed, volume, and occupancy). No additional traffic data beyond speed, volume, and occupancy will be provided in R9. The same facilities regarding protection of data (actual speed vs. speed ranges vs. no data) will be provided for the G4 RTMS as has been done previously for the X3 RTMS and (particularly) external RTMSs. See the sections below for details on portions of the TSS human-machine interface that are new or changed for R9.

### 9.1.1 Add TSS

After clicking the Add TSS link, the Add TSS form is displayed as shown. Each section of this form is discussed in the sections below.

**Add RTMS**

**General TSS Information**

Model: G4 RTMS  
Name: New Detector  
Owning Organization: SHA  
Maintaining Organization: Radio Shop  
Op Center to Receive Alerts: None  
Polling Interval (HH:MM:SS): 0 : 5 : 0  
Enable Device Logging: ☒  
Enable Scheduled Built-in Test: ☒

**Location (Edit)**

Location Description: MD  
County:  
Region:  
State: MARYLAND  
(Roadway location not specified.)  
Lat/Long: Not defined

**Zone Groups**

Grp Num	Description	Detection Zones	Direction	Posted Spd	Actions
1	Eastbound local	1, 2, 3	East	55	<a href="#">Edit</a> <a href="#">Remove</a>
2	Eastbound express	4, 5, 6	East	65	<a href="#">Edit</a> <a href="#">Remove</a>
3	Westbound express	7, 8, 9	West	65	<a href="#">Edit</a> <a href="#">Remove</a>
4	Westbound local	<input type="checkbox"/> 1 <input type="checkbox"/> 2 <input type="checkbox"/> 3 <input type="checkbox"/> 4 <input type="checkbox"/> 5 <input type="checkbox"/> 6 <input type="checkbox"/> 7 <input type="checkbox"/> 8 <input type="checkbox"/> 9 <input checked="" type="checkbox"/> 10 <input checked="" type="checkbox"/> 11 <input checked="" type="checkbox"/> 12	-- Select -- None North East South West Inner Loop Outer Loop		<a href="#">Add</a>

**Device Comm Settings**

Drop Address (0-255): 1

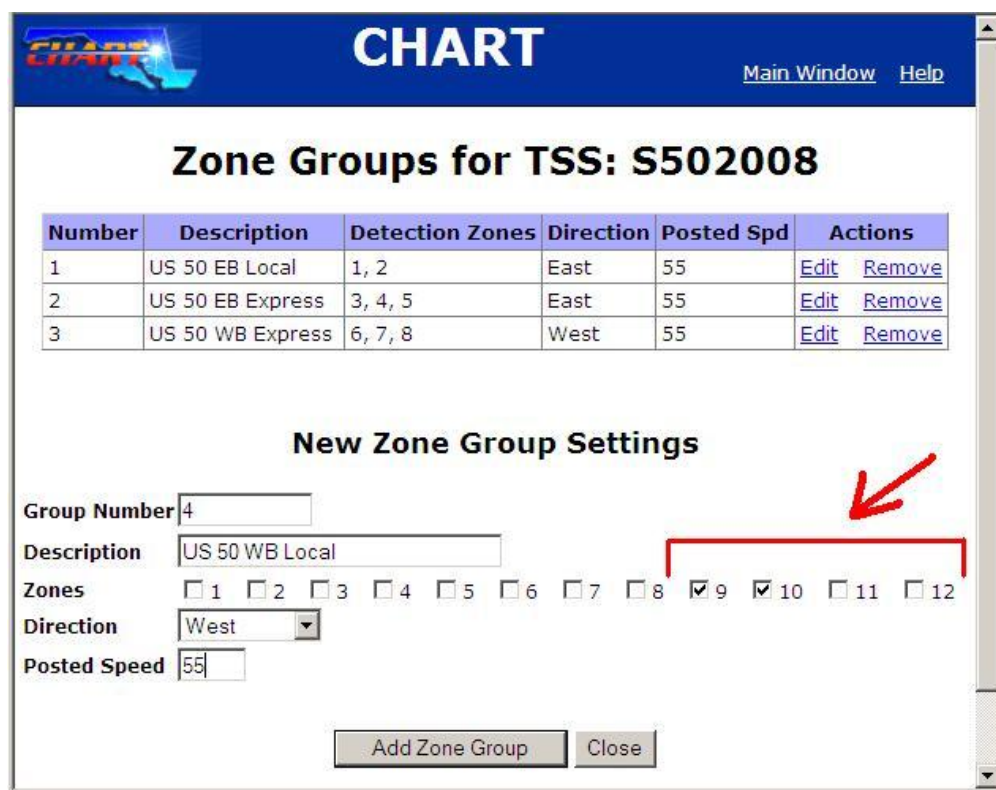
**Figure 9-1 Add TSS Form**

The Add TSS form is changed in R9 to include a model select list. The available selections are X3 RTMS and G4 RTMS. The X3 RTMS (then known as just “RTMS”) was the only TSS model supported by CHART prior to R9 and thus model selection was not needed. New for R9 is support for the G4 RTMS. When the G4 RTMS model is selected the Zone Groups can be configured to contain up to 12 zones. (Therefore, potentially up to 12 zone groups are supported, in the extreme case that each zone group would have only one zone, although in reality that is not a likely configuration.) In addition, both models support a new checkbox called “Enable Scheduled Built-in Test”.

## 9.1.2 Configure TSS

### 9.1.2.1 Configure Zone Groups

Configuring zone groups for the new G4 RTMS is identical to configuring zone groups for the existing X3 RTMS, except that up to 12 zones can be used. Editing zone groups in progress is shown below. In this figure, three zone groups using zones 1 through 8 have already been configured, and a fourth zone group comprising zones 9 and 10 are currently being configured. (Note that potentially up to 12 zone groups are supported, in the extreme case that each zone group would have only one zone, although in fact that is not a realistic configuration.)



**CHART** Main Window Help

### Zone Groups for TSS: S502008

Number	Description	Detection Zones	Direction	Posted Spd	Actions
1	US 50 EB Local	1, 2	East	55	<a href="#">Edit</a> <a href="#">Remove</a>
2	US 50 EB Express	3, 4, 5	East	55	<a href="#">Edit</a> <a href="#">Remove</a>
3	US 50 WB Express	6, 7, 8	West	55	<a href="#">Edit</a> <a href="#">Remove</a>

### New Zone Group Settings

Group Number:

Description:

Zones: ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6 ☐ 7 ☐ 8 ☒ 9 ☒ 10 ☐ 11 ☐ 12

Direction:

Posted Speed:

Figure 9-2 Edit TSS Zone Groups Form

### 9.1.2.2 Specify Whether to Run Scheduled BIT

A detector can be configured to run BIT automatically at the system-wide scheduled TSS BIT time (as configured in the system profile) or not. Check the checkbox marked “Enable Scheduled Built-in Test” to enable scheduled BIT to run each night.

CHART - Windows Internet Explorer

**CHART** [Main Window](#) [Help](#)

### Basic Settings For TSS: S315016

Model	G4 RTMS
Name	<input type="text" value="S315016"/>
Owning Organization	<input type="text" value="SHA"/>
Maintaining Organization	<input type="text" value="SHA"/>
Op Center to Receive Alerts	<input type="text" value="None"/>
Polling Interval (HH:MM:SS)	<input type="text" value="0"/> : <input type="text" value="5"/> : <input type="text" value="0"/>
Enable Device Logging	<input checked="" type="checkbox"/>
Enable Scheduled Built-in Test	<input checked="" type="checkbox"/>

**Figure 9-3 Edit TSS Basic Configuration Form**



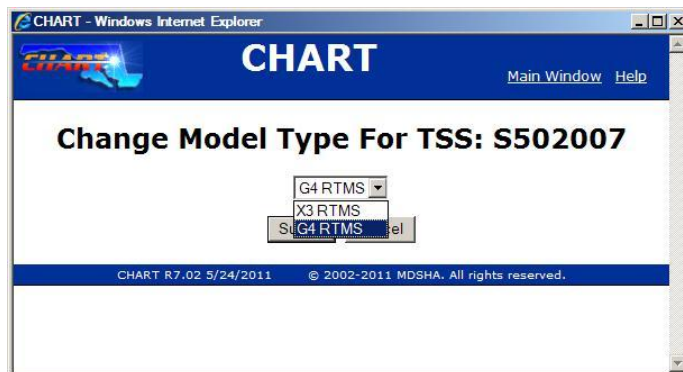
### 9.1.2.3 Change TSS Model Type

If a TSS is offline, the model type can be changed. A link labeled “change” appears when the device is offline, as shown.

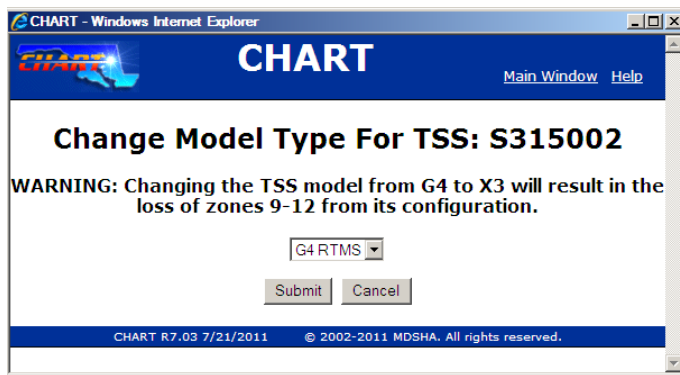


**Figure 9-4 TSS Change Model Type Request**

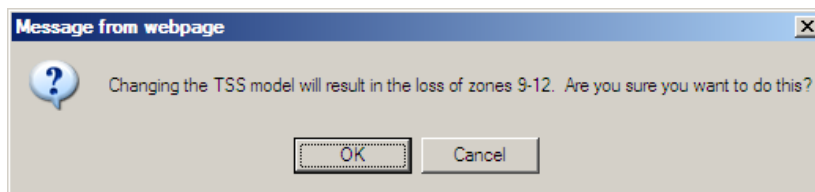
Clicking the “change” link brings up a model selection popup dialog allowing the user to change the model type of the TSS, as shown. The two choices are X3 RTMS and G4 RTMS. The current model type of the detector will be pre-selected. If the device is being changed from G4 to X3, and the G4 has zones numbered above 8, the zones above zone 8 will be removed, along with any zone groups left empty by this process. The model selection popup dialog indicates this, and a confirmation popup will be displayed to warn of this if this is going to happen.



**Figure 9-5 Change TSS Model Type Popup**



**Figure 9-6 Change TSS Model Type Popup with Warning Text**



**Figure 9-7 Change TSS Model Type Warning Dialog**

### **9.1.3 View TSS**

The TSS details page for the two model types will look identical to each other, and very similar to the existing TSS details page. If the TSS is a G4 RTMS, up to 12 lanes of data will be displayed, as shown. This illustrates 12 lanes of data, 6 lanes in each direction, for a fictitious 12-lane highway.

# Detector: S315016

I-495 EB (I/L) @ Persimmon Tree

## Status

Mode: Online

Last Reported Status: OK

Last BIT Result: OK, as of 02/28/11 10:33

Last BIT Failure: 02/19/11 03:05

Last Poll Time: 18:50

## Actions

[Take Offline / Put in Maint Mode](#)

[Get Status](#)

[Run Built-in Test](#)

[Copy RTMS](#)

## Traffic Parameters

Zone Group	Speed (MPH)	Volume	Occupancy (%)
Monitor East Bound Traffic	46.7	394	27.2
↳ Zone 1	42.2	72	1.1
↳ Zone 2	50.0	97	73.2
↳ Zone 3	59.7	80	6.6
↳ Zone 4	29.1	39	16.1
↳ Zone 5	36.7	30	11.3
↳ Zone 6	62.9	76	55.1
Monitor West Bound Traffic	21.5	424	46.7
↳ Zone 7	16.6	39	40.0
↳ Zone 8	12.4	72	71.2
↳ Zone 9	29.3	98	97.8
↳ Zone 10	28.5	99	2.1
↳ Zone 11	18.2	63	25.0
↳ Zone 12	24.3	53	44.5

## Configuration

Model: X3 RTMS (take offline to change)

Basic Settings:

Name: S315016

Network Connection Site: localhost

Owning Org: SHA

Maintaining Org: SHA

Op Center to Receive Alerts: None

Polling interval: 00:05:00


Device Logging: ON


Run Scheduled Built-in Test: OFF

Figure 9-8 TSS Details Page (top part)

In addition to up to 12 zones of data for a G4 RTMS, both detector types will also indicate the last time BIT has run, and the last time BIT failed.

Below the TSS Traffic Parameters, in the Configuration section, the GUI will provide a link to change the TSS model type if the device is offline, otherwise will indicate the device must be taken offline to change the model type. The GUI will indicate whether BIT is scheduled to run for this TSS or not. Also, the zone group configuration may indicate up to 12 zones in the zone groups if the TSS is a G4 RTMS.


**Configuration**  
**Model:** G4 RTMS (take offline to change)   
**Basic Settings:**  

<b>Name:</b>	S315016
<b>Network Connection Site:</b>	localhost
<b>Owning Org:</b>	SHA
<b>Maintaining Org:</b>	SHA
<b>Op Center to Receive Alerts:</b>	None
<b>Polling interval:</b>	00:05:00
<b>Device Logging:</b>	ON
<b>Run Scheduled Built-in Test:</b>	ON 


**Location:** [\(Edit\)](#) [\(Show on Map\)](#)  

<b>Location Description</b>	US 50 @ I-97
<b>County</b>	Anne Arundel County
<b>Region</b>	
<b>State</b>	Maryland
<b>Route Type</b>	US Route
<b>Route</b>	US 50
<b>Direction</b>	None
<b>Point Along Roadway</b>	
<b>Lat/Long</b>	38.983888° N, 76.588853° W ( Operator - dbell )

**Zone Groups:** [\(Edit Map Display Options\)](#)  
**Bearing:** 0°  
**Primary Direction Zone Groups**  

Number	Description	Detection Zones	Direction	Posted Spd	Display Direction
1	Monitor East Bound Traffic	1, 2, 3, 4, 5, 6 	East	55	Toward Bearing

  
**Opposite Direction Zone Groups**  

Number	Description	Detection Zones	Direction	Posted Spd	Display Direction
2	Monitor West Bound Traffic	7, 8, 9, 10, 11, 12 	West	55	Opposite Bearing

  
**Comm Settings:**  

<b>Drop Address:</b>	7
<b>Port Type:</b>	TCP/IP
<b>IP Address:</b>	0.0.0.0
<b>IP Port:</b>	

Figure 9-9 TSS Details Page (bottom part)

### 9.1.4 Manage TSS

There is one new action which can be performed on TSSs. There will be a new link to run Built-in Test (BIT) on a TSS (X3 or G4). This will be available in maintenance mode only, and is expected to be available to hardware maintenance personnel (e.g., Radio Shop, administrators) only. The link is displayed in the actions column as shown below.

**Detector: S403004**  
I-695 N @ I-95

**Status**  
Mode: Maintenance  
Last Reported Status: OK  
Hardware Failure Details: OK  
Last Status Time: 17:53

**Actions**  
[Put Online / Take Offline](#)  
[Get Status](#)  
[Run Built In Test](#)  
[Copy RTMS](#)

**Traffic Parameters**

Zone Group	Speed (MPH)	Volume	Occupancy (%)
Inner Loop	42.5	144	39.0
Zone 1	59.9	7	31.4

Figure 9-10 TSS Details Page – Run Built-in Test

A popup will be displayed reminding the user that running BIT stops the collection of data for 20 seconds, as shown below. If the user confirms, BIT will be run, and results will be displayed in the Command Status window.

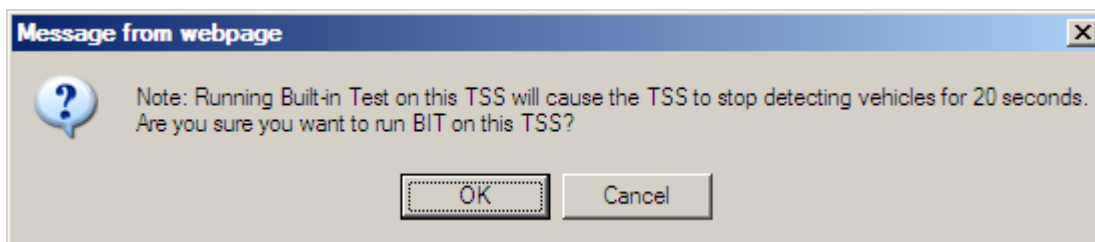


Figure 9-11 TSS Built-in Test - Confirmation

The time BIT was last run will be updated and any detected hardware faults will be listed in the Hardware Failure Details section. If the BIT comes back with any faults, the TSS operational status will be set to Hardware Failure, and if BIT comes back clean, the status will be set to OK. (If there is no response, status will be set to Comm Failure.)

## **9.2 System Interfaces**

### **9.2.1 Class Diagrams**

#### **9.2.1.1 TSSManagement (Class Diagram)**

This class diagram contains the interfaces, structs, and typedefs that are to be defined in IDL and provide the external interface to the TSSManagement package of the CHART II system.

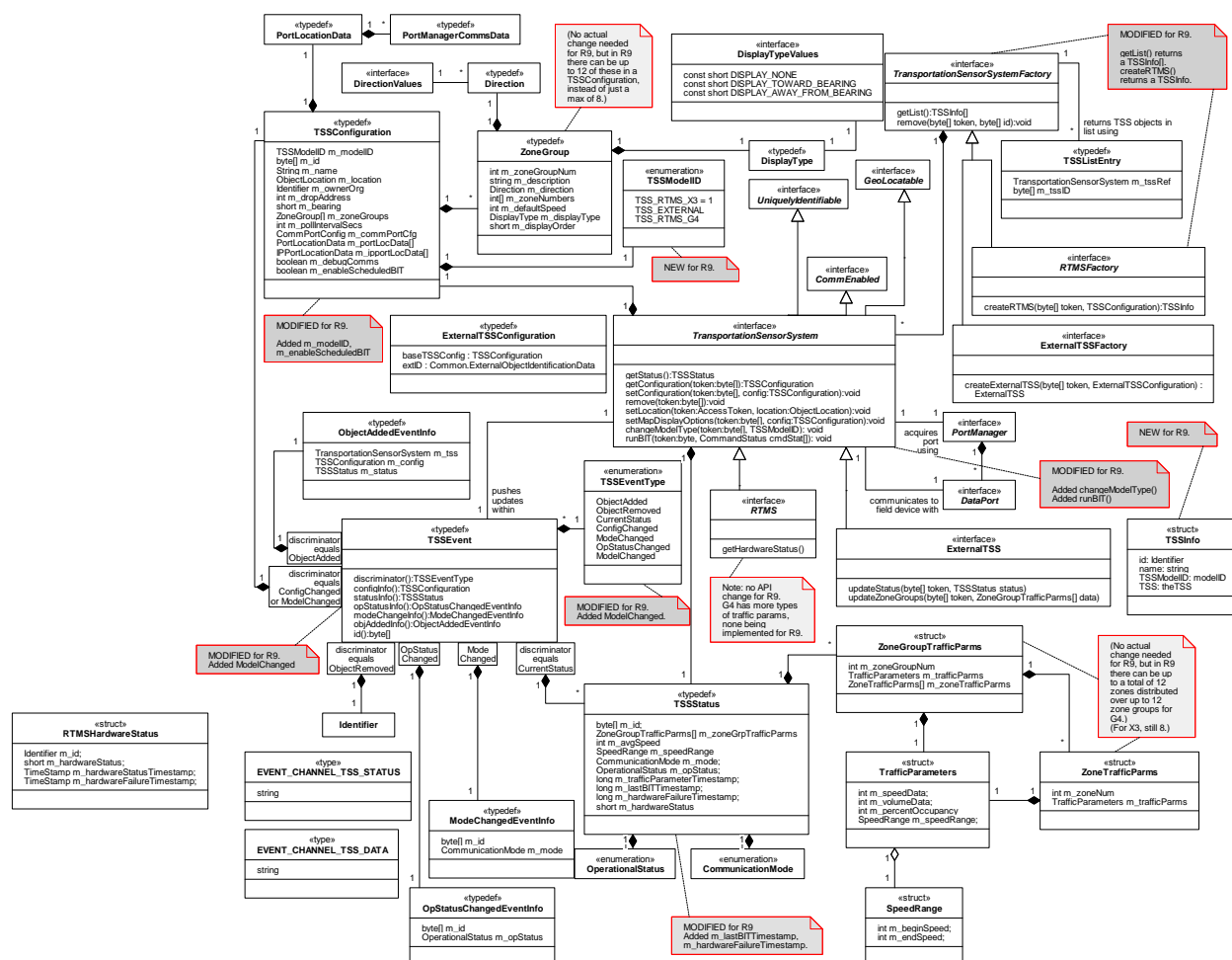


Figure 9-2. TSSManagement (Class Diagram)

#### 9.2.1.1.1 CommEnabled (Class)

The CommEnabled interface is implemented by objects that can be taken offline, put online, or put in maintenance mode through a standard interface. These states typically apply only to field devices. When a device is taken offline, it is no longer available for use through the system and automated polling (if any) is halted. When put online, a device is again available for use by TrafficEvents within the system and automated polling is enabled (if applicable). When put in maintenance mode a device is offline (i.e., cannot be used by TrafficEvents), and maintenance commands appropriate for the particular type of device are allowed to help in troubleshooting.

#### 9.2.1.1.2 CommunicationMode (Class)

The CommunicationMode class enumerates the modes of operation for a device: ONLINE, OFFLINE, and MAINT\_MODE. ONLINE is used to indicate the device is available to the operational system. OFFLINE is used to indicate the device is not available to the online system and communications to the device have been disabled. MAINT\_MODE is used to indicate that the device is available only for maintenance / repair activities and testing.



#### **9.2.1.1.3 DataPort (Class)**

A DataPort is a port that allows binary data to be sent and received. Ports of this type support a receive method that allows a chunk of all available data to be received. This method prevents callers from having to issue many receive calls to parse a device response. Instead, this receive call returns all available data received within the timeout parameters. The caller can then parse the data within a local buffer. Using this mechanism, device command and response should require only one call to send and one call to receive.

#### **9.2.1.1.4 Direction (Class)**

This type defines a short value that is used to indicate a direction of travel as defined in DirectionValues.

#### **9.2.1.1.5 DirectionValues (Class)**

This interface contains constants for directions as defined in the TMDD.

#### **9.2.1.1.6 DisplayType (Class)**

This type defines a short value that is used to indicate the display type of a zone group as defined in DisplayTypeValues. The display type indicates whether a zone group arrow will appear on the maps and if so the direction the zone group arrow will be displayed in relation to the TSS bearing (e.g. toward the bearing or away from the bearing).

#### **9.2.1.1.7 DisplayTypeValues (Class)**

This interface contains constants for display types used by zone groups. The constant include: DISPLAY\_NONE (do not display on maps), DISPLAY\_TOWARD\_BEARING (display toward the TSS bearing), and DISPLAY\_AWAY\_FROM\_BEARING (display 180 degrees away from TSS bearing).

#### **9.2.1.1.8 EVENT\_CHANNEL\_TSS\_DATA (Class)**

This is a static string that contains the name of the event channel used to push events that contain Transportation Sensor System traffic parameter data. The following TSSEventTypes are pushed on EVENT\_CHANNEL\_TSS\_DATA channels:

CurrentStatus

#### **9.2.1.1.9 EVENT\_CHANNEL\_TSS\_STATUS (Class)**

This is a static string that contains the name of the event channel used to push events relating to the change in a Transportation Sensor System status and/or configuration. The following TSSEventTypes are pushed on EVENT\_CHANNEL\_TSS\_STATUS channels:

ObjectAdded

ObjectRemoved

ConfigChanged

ModeChanged

ModelChanged

OpStatusChanged

#### **9.2.1.1.10 ExternalTSS (Class)**

This interface represents an External Systems TSS in the Chart System. I.E. a proxy for a physical TSS outside of Chart.

#### **9.2.1.1.11 ExternalTSSConfiguration (Class)**

This class holds configuration data for an ExternalTSS. It extends the TSSConfiguration data by including a reference to the base TSSConfig.

baseTSSConfig - Reference to the base TSSConfig.

extID - This objects holds the External System Name / Ext Agency / Ext id for this Extenral TSS. This uniquely identifies it in Chart.

#### **9.2.1.1.12 ExternalTSSFactory (Class)**

This interface extends the TransportationSensorSystemFactory interface to allow support of ExternalTSS objects in Chart.

#### **9.2.1.1.13 GeoLocatable (Class)**

This interface is implemented by objects that can provide location information to their users.

#### **9.2.1.1.14 Identifier (Class)**

Wrapper class for a CHART2 identifier byte sequence. This class will be used to add identifiable objects to hash tables and perform subsequent lookup operations.

#### **9.2.1.1.15 ModeChangedEventInfo (Class)**

This struct contains information pushed with a ModeChanged event.

m\_id - The ID of the TSS whose communication mode has changed.

m\_mode - The new communication mode for the TSS.

#### **9.2.1.1.16 ObjectAddedEventInfo (Class)**

This structure contains information passed in the ObjectAdded event pushed on a TSS status event channel. It contains the object reference that has been added along with its configuration values and current status values.

#### **9.2.1.1.17 OperationalStatus (Class)**

The OperationalStatus class enumerates the types of operational status a device can have: OK (normal mode), COMM\_FAILURE (no communications to the device), or

HARDWARE\_FAILURE (device is reachable but is reporting a hardware failure).

#### **9.2.1.1.18 OpStatusChangedEventInfo (Class)**

This struct contains data passed with an OpStatusChanged event.

m\_id - The ID of the TSS whose operational status has changed.

m\_opStatus - The new operational status for the device.

#### **9.2.1.1.19 PortLocationData (Class)**

This class contains configuration data that specifies the communication server(s) to use to communicate with a device.

m\_commsData - One or more objects identifying the communications server (PortManager) to use to communicate with the device, in order of preference.

m\_portType - The type of port to use to communicate with the device (ISDN modem, POTS modem, direct, etc.)

m\_portWaitTimeSecs - The maximum number of seconds to wait when attempting to acquire a port from a port manager.

#### **9.2.1.1.20 PortManager (Class)**

A PortManager is an object that manages shared access to communications port resources. The getPort method is used to request the use of a port from the PortManager. Requests for ports specify the type of port needed, the priority of the request, and the maximum time the requester is willing to wait if a port is not immediately available. When the port manager returns a port, the requester has exclusive use of the port until the requester releases the port back to the PortManager or the PortManager reclaims the port due to inactivity.

#### **9.2.1.1.21 PortManagerCommsData (Class)**

This class contains values that identify a port manager and the phone number to dial to access a device from the given port manager. This class exists to allow for the phone number used to access a device to differ based on the port manager to take into account the physical location of the port manager within the telephone network. For example, when dialing a device from one location the call may be long distance but when dialing from another location the call may be local.

#### **9.2.1.1.22 RTMS (Class)**

The Remote Traffic Microwave Sensor (RTMS) is a detector manufactured by EIS, Inc. capable of providing lane level volume, speed, and occupancy data for up to 8 or 12 lanes of a roadway at a single location, depending on RTMS model. This interface serves to identify TransportationSensorSystem objects as being of the type RTMS. It also provides a place holder for future operations that may not apply to TSS objects in general and are instead RTMS specific.

#### **9.2.1.1.23 RTMSFactory (Class)**

Objects which implement RTMSFactory are capable of adding an RTMS to the system.

#### **9.2.1.1.24 SpeedRange (Class)**

This struct is used to specify a speed range. The speed range is defined in MPH and has an upper and lower limit inclusive. Note: m\_endSpeed of zero means range is > m\_beginSpeed. MPH is implied.

#### **9.2.1.1.25 TrafficParameters (Class)**

This struct contains traffic parameters that are sensed and reported by a Traffic Sensor System such as the RTMS.

m\_speedData - The arithmetic mean of the speeds collected over a sample period in miles per hour in tenths. (thus 550 == 55.0 MPH) Valid values are 0 to 2550. A value of 65535 is used to indicate a missing or invalid value (such as when the volume for the sample period is zero).

m\_volumeData - The count of vehicles for the sample period. Valid values 0 to 65535. A value of 65535 represents a missing value.

m\_percentOccupancy - The percentage of occupancy of the roadway in tenths of a percent. (thus 1000 = 100.0 percent). Valid values are 0 to 1000. A value of 65535 represents a missing or invalid value.

#### **9.2.1.1.26 TransportationSensorSystem (Class)**

A Transportation Sensor System (TSS) is a generic term used to describe a class of technology used for detection within the transportation industry. Examples of TSS devices range from the advanced devices, such as RTMS, to basic devices, such as single loop detectors.

This software interface is implemented by objects that provide access to the traffic parameters sensed by a Transportation Sensor System. Transportation Sensor Systems are capable of providing detection for one or more detection zones. A single loop detector would have one detection zone, while an RTMS could have up to 12 detection zones.

#### **9.2.1.1.27 TransportationSensorSystemFactory (Class)**

This interface is implemented by objects that are used to create and serve TransportationSensorSystem (TSS) Objects. All factories of TSS objects can return the list of TSS objects which they have created and serve. Derived interfaces are used to provide factories to create specific make, models, and types of TransportationSensorSystem objects.

#### **9.2.1.1.28 TSSConfiguration (Class)**

This class holds configuration data for a transportation sensor system (TSS) as follows:

m\_id - The unique identifier for this TSS. This field is ignored when the object is passed to the TSS to change its configuration.

m\_name - The name used to identify the TSS.

m\_location - A descriptive location of the TSS.

m\_dropAddress - The drop address for the device.

m\_bearing - The bearing in degrees for displaying the TSS on the map. Valid values are from -1 to 359 (-1 = bearing not defined, 0 = East, 90 = North, 180 = West, and 270 = South). The default value is -1.

m\_zoneGroups - Logical groupings of detection zones, used to provide a single set of traffic parameters for one or more detection zones.

m\_pollIntervalSecs - The interval on which the TSS should be polled for its current traffic parameters (in seconds).

m\_enableScheduledBIT - Flag that determines if the device scheduled built-in test is enabled to run (Boolean).

m\_commPortCfg - Communication configuration values.

m\_portLocData - Configuration information that determines which port manager(s) should be used to establish a connection with the SensorSystem.

m\_debugComms - Flag used to enable/disable the logging of communications data for this TSS. When enabled, command and response packets exchanged with the device are logged to a debugging log file.

#### **9.2.1.1.29 TSSEvent (Class)**

This class is a CORBA union that contains varying data depending on the current value of the discriminator.

If the discriminator is ConfigChanged, this union contains a TSSConfiguration object.

If the discriminator is ObjectAdded, this union contains an ObjectAddedEventInfo object.

If the discriminator is ObjectRemoved, this union contains a byte[] containing the unique identifier for the Traffic Sensor System that was removed.

If the discriminator is CurrentStatus the union contains an array of one or more TSSStatus objects.

If the discriminator is ModeChanged, the union contains a ModeChangedEventInfo.

If the discriminator is ModelChanged, the union contains a TSSConfiguration object.

If the discriminator is OpStatusChanged, the union contains an OpStatusChangedEventInfo object.

#### **9.2.1.1.30 TSSEventType (Class)**

This enumeration defines the types of events that may be pushed on an event channel by a Transportation Sensor Status object. The values in this enumeration are used as the discriminator in the TSSEvent union.

ObjectAdded - a TransportationSensorSystem has been added to the system.

ObjectRemoved - a TransportationSensorSystem has been removed from the system.

CurrentStatus - The event contains the current status of one or more Transportation Sensor System objects.

ConfigChanged - One or more configuration values for the Transportation Sensor System have been changed.

ModeChanged - The communications mode of the TransportationSensorSystem has changed.

ModelChanged – The model type for the Transportation Sensor System has changed.

OpStatusChanged - The operational status of the TransportationSensorSystem has changed.

#### **9.2.1.1.31 TSSInfo (Class)**

This struct contains information about a single TSS.

#### **9.2.1.1.32 TSSListEntry (Class)**

This struct is used to pass a TransportationSensorSystem object together with its ID. This struct is provided for convenience because when discovering an object, it is usually required to make a call to the object's getID() method.

#### **9.2.1.1.33 TSSModelID (Class)**

This enumeration enumerates the supported TSS hardware models.

#### **9.2.1.1.34 TSSStatus (Class)**

This class holds current status information for a TSS as follows:

m\_id - The ID of the TSS for which this status applies.

m\_zoneGrpTrafficParms - The traffic parameters for each ZoneGroup of the Transportation Sensor System as specified in the Sensor system's TSSConfiguration object.

m\_mode - The communication mode of the TSS.

m\_opStatus - The operational status for the TSS.

M\_lastBITTimestamp - A timestamp that records when the last device BIT was executed.

m\_trafficParameterTimestamp - A timestamp that records when the traffic parameter data was collected from the device.

m\_avgSpeed - average speed at the detector level.

m\_speedRange - speed range at the detector level (avg speed).

#### **9.2.1.1.35 UniquelyIdentifiable (Class)**

This interface will be implemented by all classes which are to be identifiable within the system. The identifier must be generated by the IdentifierGenerator to ensure uniqueness.

#### **9.2.1.1.36 ZoneGroup (Class)**

This class is used to group one or more detection zones of a Transportation Sensor System into a logical grouping. Traffic parameters for all detection zones included in the group are averaged to provide a single set of traffic parameters for the group.

#### **9.2.1.1.37 ZoneGroupTrafficParms (Class)**

This struct contains traffic parameters for a ZoneGroup.

m\_zoneGroupNumber - The number of the zone group for which the traffic parameters apply.

m\_trafficParms - The traffic parameter values for the zone group.

m\_zoneTrafficParms - zone parms for each zone in the group.

#### **9.2.1.1.38 ZoneTrafficParms (Class)**

This struct contains traffic parameters for a Zone.

m\_zoneNumber - The number of the zone for which the traffic parameters apply.

m\_trafficParms - The traffic parameter values for the zone.

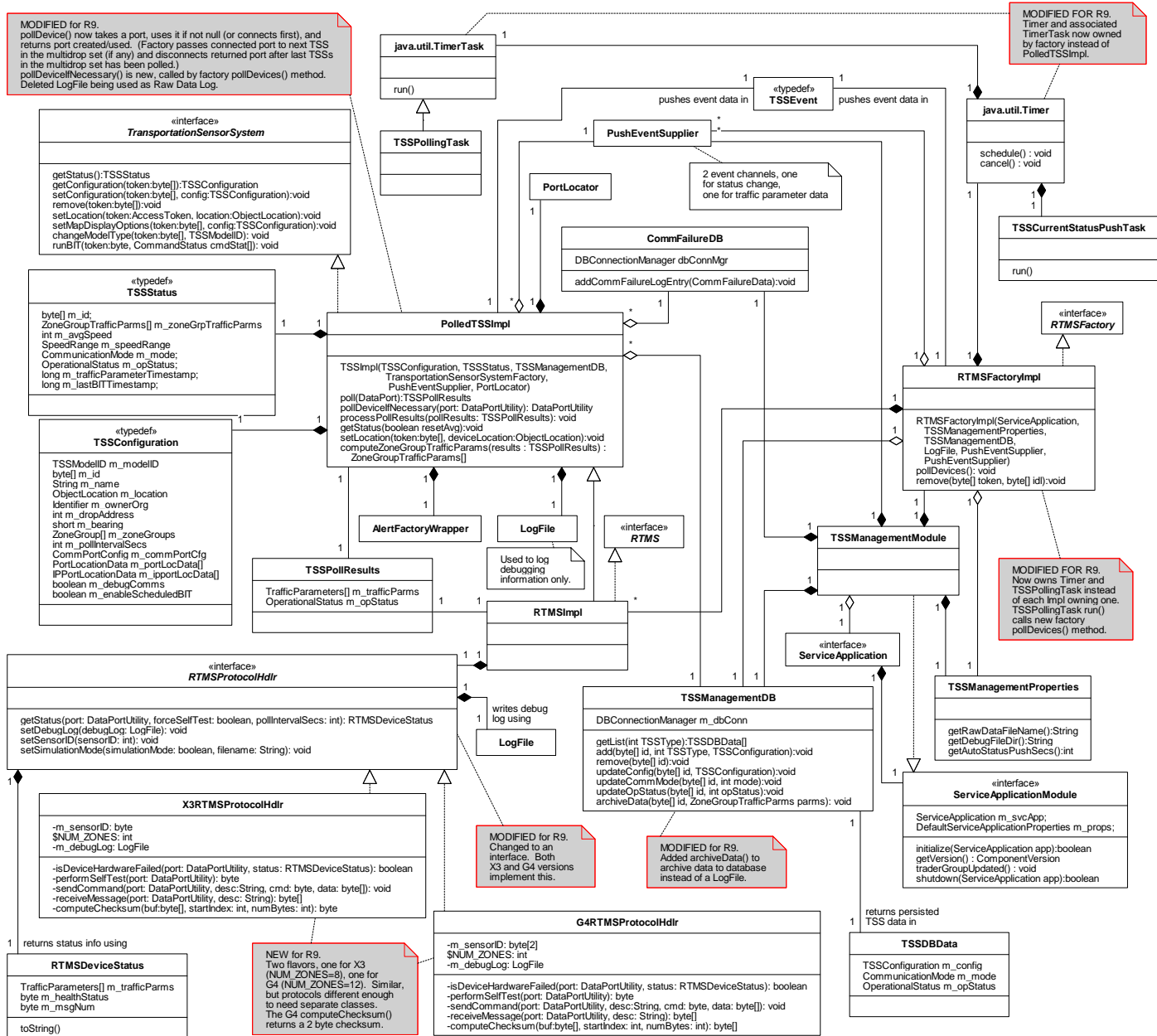
## **9.3 TSSManagementPkg**

### **9.3.1 Class Diagrams**

#### **9.3.1.1 TSSManagementModulePkg (Class Diagram)**

This package manages all server activities related to Traffic Sensor Systems. Currently only Remote Traffic Microwave Sensor (RTMS) type devices are supported however it is designed to handle other TSS devices types. Devices are periodically polled (responding to a device-created hardware status event is not supported) and results are reported on CORBA event channels.





**Figure 9-3. TSSManagementModulePkg (Class Diagram)**

#### 9.3.1.1.1 AlertFactoryWrapper (Class)

This singleton class provides a wrapper for the Alert Factory that provides automatic location of an Alert Factory and automatic re-discovery should the Alert Factory reference return an error. This class also allows for built-in fault tolerance by automatically failing over to a "working" Alert Factory without the user of this class being aware that this being done. In addition, this class defers the discovery of the Alert Factory until its first use, thus eliminating a start-up dependency for modules that use the Alert Factory.

This class delegates all of its method calls to the system AlertFactory using its currently known good reference to an AlertFactory. If the current reference returns a CORBA failure in the delegated call, this class automatically switches to another reference. When there are no good references (as is true the first time the object is used), this class issues a trader query to (re)discover the published Alert Factory objects in the system. During a method call, the trader will be queried at most one time and under normal circumstances, not at all.

#### **9.3.1.1.2 CommFailureDB (Class)**

This class is a utility used to log an entry in the Comm Failure log table in the database. This table is used to log details about any comm failure that occurs in the system.

#### **9.3.1.1.3 G4RTMSProtocolHdlr (Class)**

This Class encapsulates the communication protocol of the G4 RTMS device. All methods that interact with the G4 RTMS use the protocol specific to the G4 RTMS to send commands and receive result messages. Any calculations and analysis of RTMS data is done using masks, command size and other G4 specific information.

#### **9.3.1.1.4 java.util.Timer (Class)**

This class provides asynchronous execution of tasks that are scheduled for one-time or recurring execution.

#### **9.3.1.1.5 java.util.TimerTask (Class)**

This class is an abstract base class which can be scheduled with a timer to be executed one or more times.

#### **9.3.1.1.6 LogFile (Class)**

This class creates a flat file for writing system trace log messages and purges them at user specified interval. The log files created by this class are used for system debugging and maintenance only and are not to be confused with the system operations log which is modeled by the OperationsLog class.

#### **9.3.1.1.7 PolledTSSImpl (Class)**

This object implements the Transportation Sensor System interface as defined in IDL. This implementation provides the base functionality required for Transportation Sensor Systems that are polled periodically to retrieve traffic parameters. The only requirement for derived classes is to provide an implementation of the abstract poll method, which communicates over a previously connected Port to obtain the traffic parameters from a TSS.

This implementation periodically polls the field device using the derived class implementation of the poll method. This implementation provides services such as raw data logging, averaging/summation of data into configured zone groups, asynchronous notification of configuration changes, and persistence/depersistence.

A DeviceFailure alert is created each time the device transitions into `HARDWARE_FAILURE`. Devices that cycle in and out of `HARDWARE_FAILURE` will send multiple DeviceFailure alerts so it is up to the AlertModule to prevent duplicate open

DeviceFailure alerts for the same device.

#### **9.3.1.1.8 PortLocator (Class)**

The PortLocator is a utility class that helps one to connect to the port used by the device. The actual implementation of the operations is done by the derived classes depending on what protocol is used for communication.

#### **9.3.1.1.9 PushEventSupplier (Class)**

This class provides a utility for application modules that push events on an event channel. The user of this class can pass a reference to the event channel factory to this object. The constructor will create a channel in the factory. The push method is used to push data on the event channel. The push method is able to detect if the event channel or its associated objects have crashed. When this occurs, a flag is set, causing the push method to attempt to reconnect the next time push is called. To avoid a supplier with a heavy supply load from causing reconnect attempts to occur too frequently, a maximum reconnect interval is used. This interval specifies the quickest reconnect interval that can be used. The push method uses this interval and the current time to determine if a reconnect should be attempted, thus reconnects can be throttled independently of a supplier's push rate.

#### **9.3.1.1.10 RTMS (Class)**

The Remote Traffic Microwave Sensor (RTMS) is a detector manufactured by EIS, Inc. capable of providing lane level volume, speed, and occupancy data for up to 8 lanes of a roadway at a single location. This interface serves to identify TransportationSensorSystem objects as being of the type RTMS. It also provides a place holder for future operations that may not apply to TSS objects in general and are instead RTMS specific.

#### **9.3.1.1.11 RTMSDeviceStatus (Class)**

This class is used to pass raw data retrieved from the RTMS to the caller of the RTMSProtocolHdlr getStatus() method.

m\_trafficParameters - the traffic parameters sensed by the device, such as volume, speed, and occupancy.

m\_healthStatus - The health status byte reported from the RTMS. A value other than 10, 20, 30, 40, 50, 60, or 70 indicates a hardware problem. (This is populated only to the X3 RTMS. The G4 RTMS does not populate its health status byte.)

m\_msgNum - The message number reported by the RTMS. This number is incremented sequentially when the RTMS dumps averaged data to a retrieval area at the end of a message period. It can be used to determine if the device is being polled too frequently or infrequently.

#### **9.3.1.1.12 RTMSFactory (Class)**

Objects which implement RTMSFactory are capable of adding an RTMS to the system.

#### **9.3.1.1.13 RTMSFactoryImpl (Class)**

This class implements the RTMSFactory interface as defined in the IDL. It holds all

RTMSImpl objects that have been created within an instance of the RTMSManagementModule and allows for the addition and removal of RTMS objects. It also allows one to query all RTMS objects currently served from the factory.

This factory contains a timer that periodically fires, causing the RTMSFactoryImpl to collect the current status of each RTMSImpl and push the collective status in a single CORBA event.

#### **9.3.1.1.14 RTMSImpl (Class)**

This class is a derivation of the PolledTSSImpl that provides functionality for obtaining the current traffic parameters from an RTMS device. It makes use of an RTMSProtocolHandler to perform the device specific protocol to obtain the traffic parameters. It moves the data from the device specific format to the generic TSSPollResults object to allow the PolledTSSImpl to combine/average data based on zone group configuration, perform raw data logging, and other services that are common to Transportation Sensor System objects.

#### **9.3.1.1.15 RTMSProtocolHdlr (Class)**

This class is a utility that encapsulates the communication protocol of the RTMS device. It provides a high level method to get the status as an object. It formats a command and sends it to the device and receives and interprets the response from the device, passing the data back to the caller in the form of an RTMSDeviceStatus object.

#### **9.3.1.1.16 ServiceApplication (Class)**

This interface is implemented by objects that can provide the basic services needed by a ChartII service application. These services include providing access to basic CORBA objects that are needed by service applications, such as the ORB, POA, Trader, and Event Service.

#### **9.3.1.1.17 ServiceApplicationModule (Class)**

This interface is implemented by modules that serve CORBA objects. Implementing classes are notified when their host service is initialized and when it is shutdown. The implementing class can use these notifications along with the services provided by the invoking ServiceApplication to perform actions such as object creation and publication.

#### **9.3.1.1.18 TransportationSensorSystem (Class)**

A Transportation Sensor System (TSS) is a generic term used to describe a class of technology used for detection within the transportation industry. Examples of TSS devices range from the advanced devices, such as RTMS, to basic devices, such as single loop detectors.

This software interface is implemented by objects that provide access to the traffic parameters sensed by a Transportation Sensor System. Transportation Sensor Systems are capable of providing detection for one or more detection zones. A single loop detector would have one detection zone, while an RTMS could have up to 12 detection zones.

#### **9.3.1.1.19 TSSConfiguration (Class)**

This class holds configuration data for a transportation sensor system (TSS) as follows:

**m\_id** - The unique identifier for this TSS. This field is ignored when the object is passed to the TSS to change its configuration.

**m\_name** - The name used to identify the TSS.

**m\_location** - A descriptive location of the TSS.

**m\_dropAddress** - The drop address for the device.

**m\_bearing** - The bearing in degrees for displaying the TSS on the map. Valid values are from -1 to 359 (-1 = bearing not defined, 0 = East, 90 = North, 180 = West, and 270 = South). The default value is -1.

**m\_zoneGroups** - Logical groupings of detection zones, used to provide a single set of traffic parameters for one or more detection zones.

**m\_pollIntervalSecs** - The interval on which the TSS should be polled for its current traffic parameters (in seconds).

**m\_enableScheduledBIT** - Flag that determines if the device scheduled built-in test is enabled to run (Boolean).

**m\_commPortCfg** - Communication configuration values.

**m\_portLocData** - Configuration information that determines which port manager(s) should be used to establish a connection with the SensorSystem.

**m\_debugComms** - Flag used to enable/disable the logging of communications data for this TSS. When enabled, command and response packets exchanged with the device are logged to a debugging log file.

#### **9.3.1.1.20 TSSCurrentStatusPushTask (Class)**

This class is a timer task that is executed on a regular interval. When this task is run, it calls into the RTMSFactoryImpl object to have it collect the status for all RTMSImpl objects and to push a CurrentStatus event with the collected data.

#### **9.3.1.1.21 TSSDBData (Class)**

This class holds data that is retrieved from the database during start-up for a Transportation Sensor System object that existed in the system during a prior run of the software.

#### **9.3.1.1.22 TSSEvent (Class)**

This class is a CORBA union that contains varying data depending on the current value of the discriminator.

If the discriminator is ConfigChanged, this union contains a TSSConfig object.

If the discriminator is `ObjectAdded`, this union contains an `ObjectAddedEventInfo` object.

If the discriminator is `ObjectRemoved`, this union contains a `byte[]` containing the unique identifier for the Traffic Sensor System that was removed.

If the discriminator is `CurrentStatus` the union contains an array of one or more `TSSStatus` objects.

If the discriminator is `ModeChanged`, the union contains a `ModeChangedEventInfo`.

If the discriminator is `ModelChanged`, the union contains a `TSSConfiguration` object.

If the discriminator is `OpStatusChanged`, the union contains an `OpStatusChangedEventInfo` object.

#### **9.3.1.1.23 TSSGroupInfo (Class)**

The `TSSGroupInfo` class is a data holder TSS data related to a group of TSSs located at a common site and using a common connection. Contains a command queue and connected port info used by all TSSs in the group. It also has functions that make port connections when needed and disconnect the port connection when not needed.

#### **9.3.1.1.24 TSSManagementDB (Class)**

This class is a utility that provides methods for adding, removing, and updating database data pertaining to Transportation Sensor Systems. Because this class is designed to be generic and work for RTMS as well as other TSS derived objects, the add method requires a model id to be passed. This allows data for a specific model to be retrieved by model specific factories during system initialization.

#### **9.3.1.1.25 TSSManagementModule (Class)**

This class provides the server side support for Transportation Sensor System objects. It creates objects needed by other classes in the module and implements the `ServiceApplicationModule` interface so it can be hosted in a `Chart2Service` application. It creates factories which in turn serve TSS objects that provide a software interface to TSS field devices.

#### **9.3.1.1.26 TSSManagementProperties (Class)**

This class provides a wrapper to the application's properties file that provides easy access to the properties specific to the `TSSManagementModule`. These properties include the name of the file where raw traffic parameter data is to be logged, the directory where debug log files are to be kept, and the interval at which the status of all TSS objects is to be collected and pushed in a CORBA event.

#### **9.3.1.1.27 TSSPollingTask (Class)**

This class is a `TimerTask` that is used by an RTMS to schedule its asynchronous polling with a `Timer` object.

#### **9.3.1.1.28 TSSPollResults (Class)**

This class is a data holder used to pass the results of device polling from the PolledTSSImpl derived class back to the base class for processing. The traffic parameter data passed is lane (detection zone) level. The operational status is the status as determined by the derived class.

m\_trafficParms - An array of traffic parameters for the current poll cycle, with one array entry for each detection zone of the device.

m\_opStatus - The operational status as determined by the derived class.

#### **9.3.1.1.29 TSSStatus (Class)**

This class holds current status information for a TSS as follows:

m\_id - The ID of the TSS for which this status applies.

m\_zoneGrpTrafficParms - The traffic parameters for each ZoneGroup of the Transportation Sensor System as specified in the Sensor system's TSSConfiguration object.

m\_mode - The communication mode of the TSS.

m\_opStatus - The operational status for the TSS.

m\_lastBITTimestamp - A timestamp that records when the device BIT was executed.

m\_trafficParameterTimestamp - A timestamp that records when the traffic parameter data was collected from the device.

m\_avgSpeed - average speed at the detector level.

m\_speedRange - speed range at the detector level (avg speed).

#### **9.3.1.1.30 X3RTMSProtocolHdlr (Class)**

This Class encapsulates the communication protocol of the X3 RTMS device. All of the methods that interact with the X3 RTMS use the protocol specific to the X3 RTMS to send commands and receive result messages. Any calculations and analysis of RTMS data is done using masks, command size and other X3 specific information.

## 9.3.2 Sequence Diagrams

### 9.3.2.1 PolledTSSImpl:changeModelType (Sequence Diagram)

A user with the proper functional rights can change the model type of a Transportation Sensor System. The previous model type value is used to detect if the model type has been changed. If the model type has changed, a new protocol handler of the type indicated by the model ID is created to replace the old protocol handler. The model id is updated in the CHART DB TSS table. Also an entry is made in the operations log to record the user has changed the model type. A CORBA event is pushed on the Status event channel to provide notification of the configuration change to other applications.

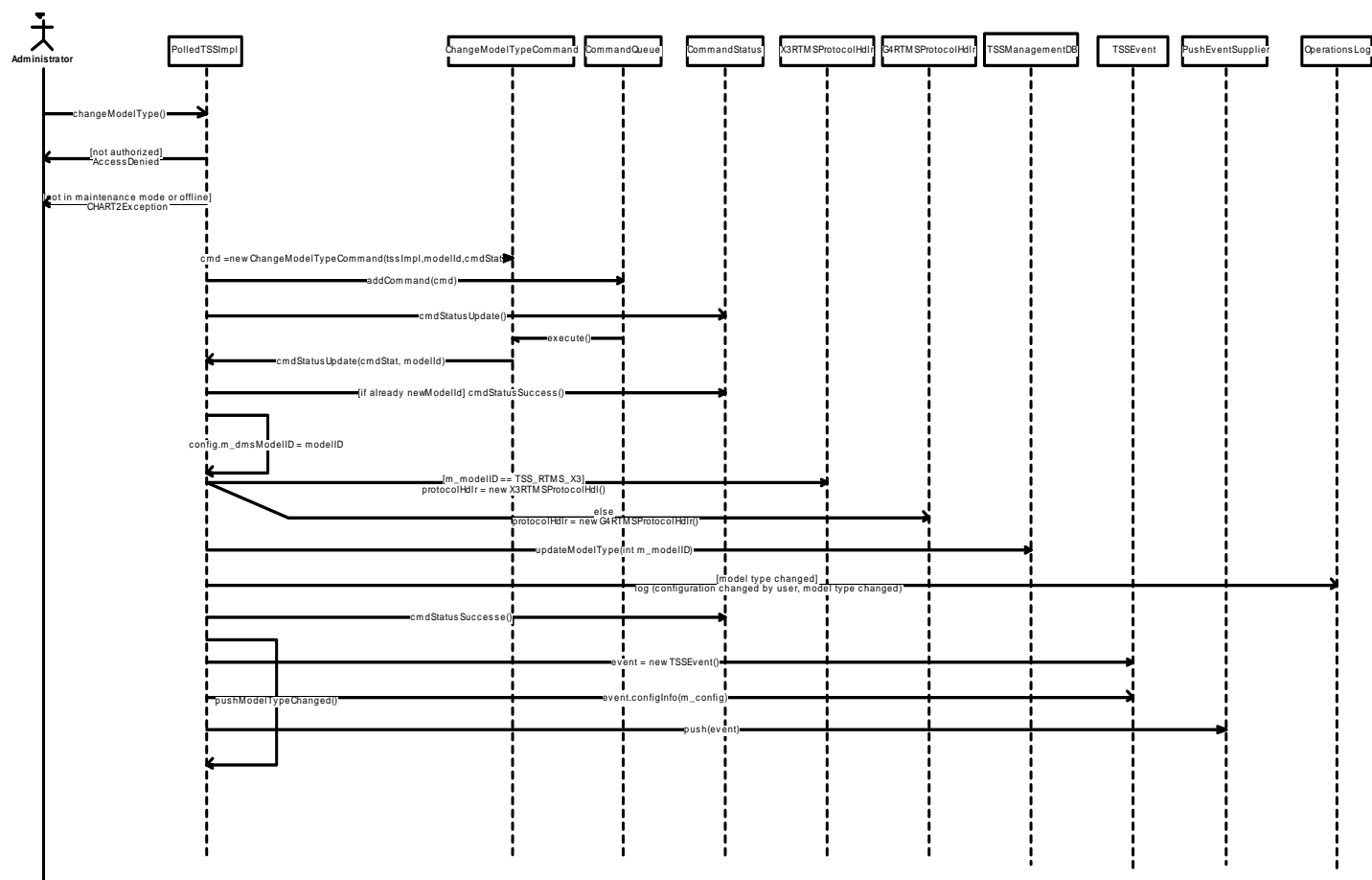
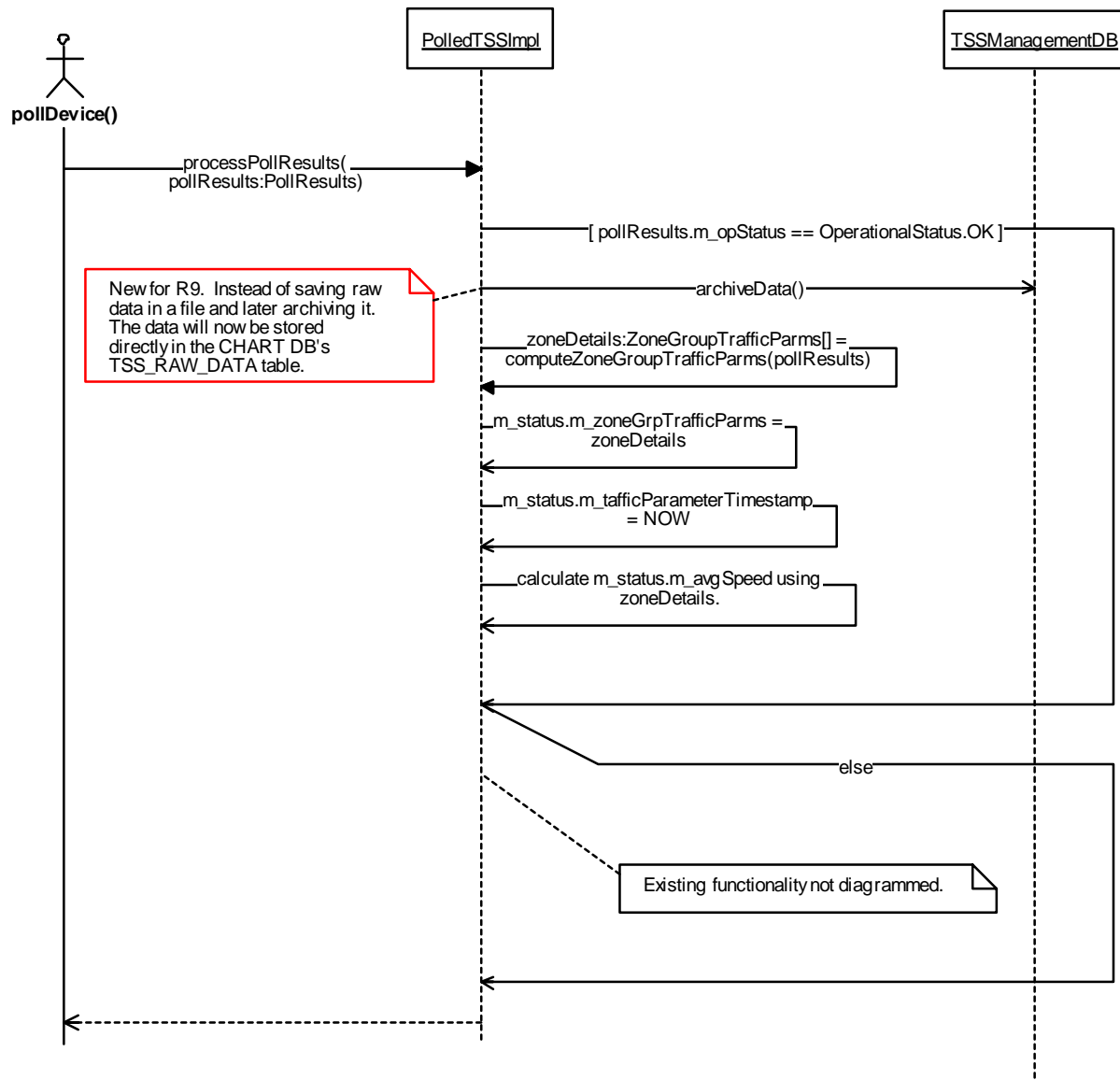


Figure 9-4. PolledTSSImpl:changeModelType (Sequence Diagram)



### 9.3.2.2 PolledTSSImpl:processPollResults (Sequence Diagram)

This method processes the TSSPollResults returned from the protocol handler as the result of a poll. If the TSSPollResults indicates the op status is OK, the following processing is done. First, raw zone level detail is stored on the CHART DB TSS\_RAW\_DATA table using the archiveData() function. Then the computeZoneGroupTrafficParms() method is called and returns an array of ZoneGroupTrafficParms. The returned array is used to update the current TSSStatus. Then the zone group data is used to calculate and average speed (TSSStatus.m\_avgSpeed). Previously the average speed was calculated on the GUI.



**Figure 9-5. PolledTSSImpl:processPollResults (Sequence Diagram)**

### 9.3.2.3 RTMSFactoryImpl:constructor (Sequence Diagram)

When the RTMSFactoryImpl is constructed, it obtains persisted data for each previously existing RTMS from the database and constructs RTMSImpl objects using this data. It creates an ordered TreeMap<String,Vector<RTMSImpl>>. Each RTMSImpl is stored in a vector within the map using the communication settings string as the key. The key is the IP + Port pair, the phone number, or serial port. The connection string is the key to a vector with RTMSs that have same connection location. As the function is reading from the database, it will create the vector for each unique connection string it encounters the first time. It then creates a TSSGroupInfo and a CommandQueue. The CommandQueue is set in the TSSGroupInfo and the TSSGroupInfo is set in the current RTMSImpl. The current RTMSImpl is added to the vector. As the next RTMSImpl is encountered with the same connection string, the function will get the vector corresponding to the connection string and use the first RTMSImpl in the vector to get the TSSGroupInfo of the group and store it in the current RTMSImpl. The current impl is then stored in the vector. This process continues until all RTMSImpl have been added to a vector and thus a TSSGroupInfo. The factory creates a TSSPollingTask that is used to poll each RTMS at scheduled intervals. Each RTMSImpl object is connected to the ORB and registered in the CORBA trading service. The factory creates a timer that is used to cause it to periodically collect the status of all RTMS objects and push the data as a CORBA event.

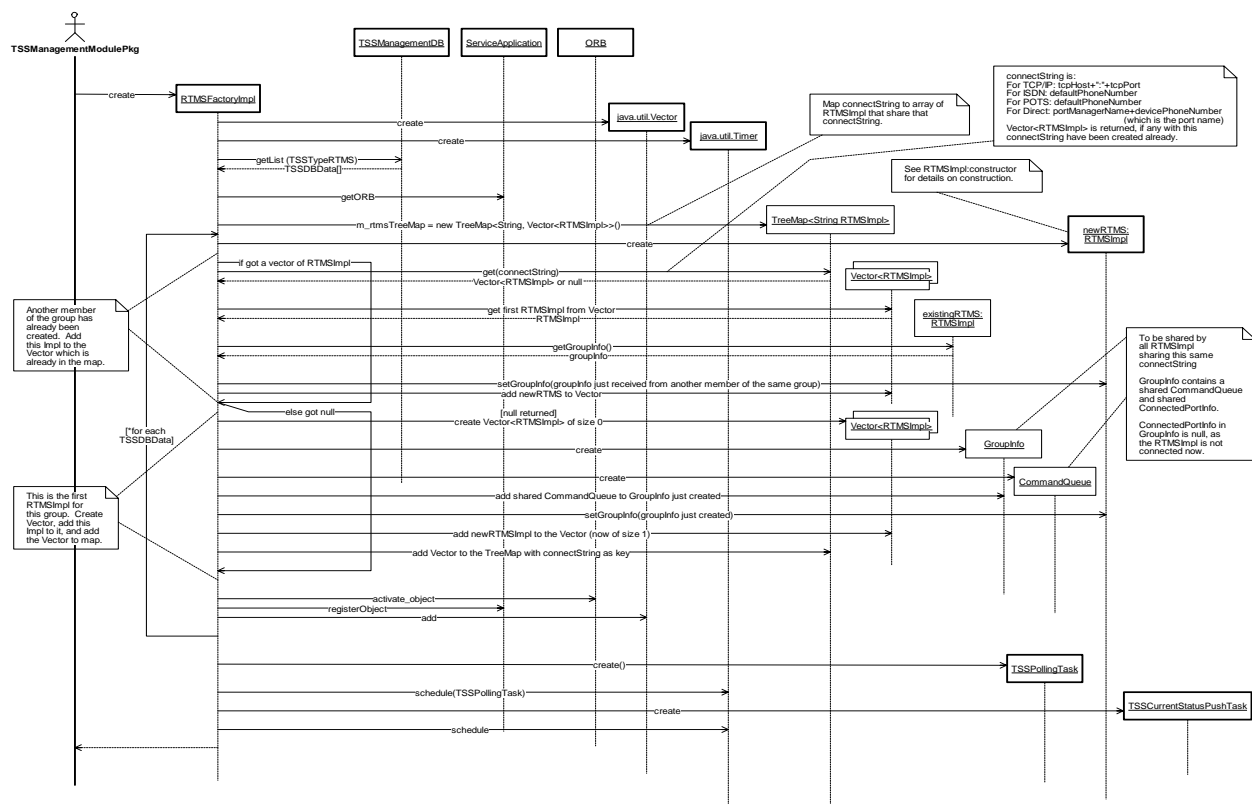


Figure 9-6. RTMSFactoryImpl:constructor (Sequence Diagram)

### 9.3.2.4 RTMSFactoryImpl:createRTMS (Sequence Diagram)

A user with the proper functional rights can add an RTMS to the system. The RTMSFactoryImpl is called with configuration data for the RTMS to be added. The RTMSFactoryImpl adds the configuration data to the database, using status information indicating the device is offline and OK. An RTMSImpl object is created using this same data and the object is added to the list of RTMSImpl objects managed by the factory. The new RTMSImpl creates either an X3 RTMS or G4 RTMS protocol handler based on the model ID contained in the configuration. If the model ID is 0 an X3 RTMS protocol handler is created. If the model ID is 1 a G4 RTMS protocol handler is created. The function then uses the RTMS connection string to get vector from the RTMS TreeMap containing vectors of RTMSImpl. The TreeMap was created using the IP +port, phone # and serial port as keys. If the map returns a vector the function gets the TSSGroupInfo from the first RTMSImpl in the vector and adds that TSSGroupInfo to the new RTMSImpl. It then adds the new RTMSImpl to the vector. If the TreeMap returns null the function creates a new Vector, CommandQueue, ConnectedPortInfo, and TSSGroupInfo. It adds the CommandQueue and ConnectedPortInfo to the TSSGroupInfo, adds the TSSGroupInfo to the new RTMSImpl, and adds the RTMSImpl to the vector. The new RTMSImpl object is connected to the ORB and published in the CORBA Trading Service. A CORBA event is pushed to allow other applications to be notified of the existence of the RTMS object.

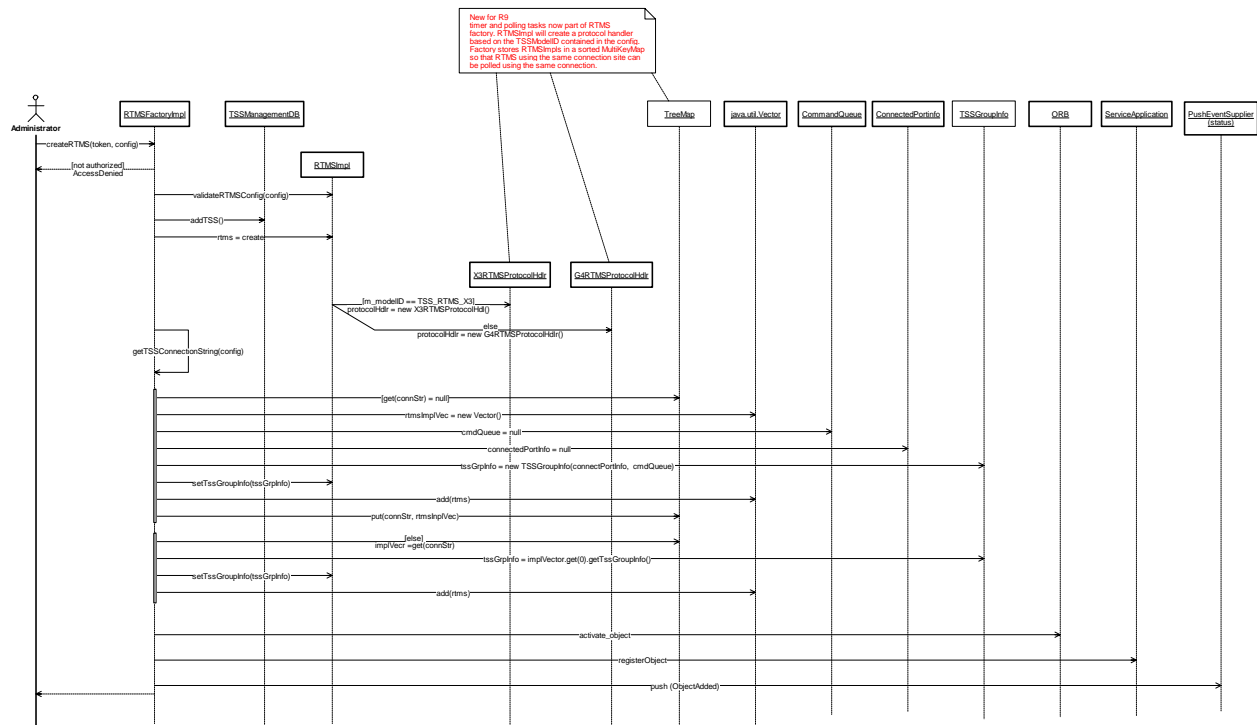


Figure 9-7. RTMSFactoryImpl:createRTMS (Sequence Diagram)

### 9.3.2.5 RTMSFactoryImpl:CurrentStatusPush (Sequence Diagram)

The RTMSFactoryImpl contains a timer used to periodically push the current status of all sensors managed by the factory. The factory retrieves the status of each RTMS and bundles all status into a single CORBA event. This event is pushed on the Data event channel.

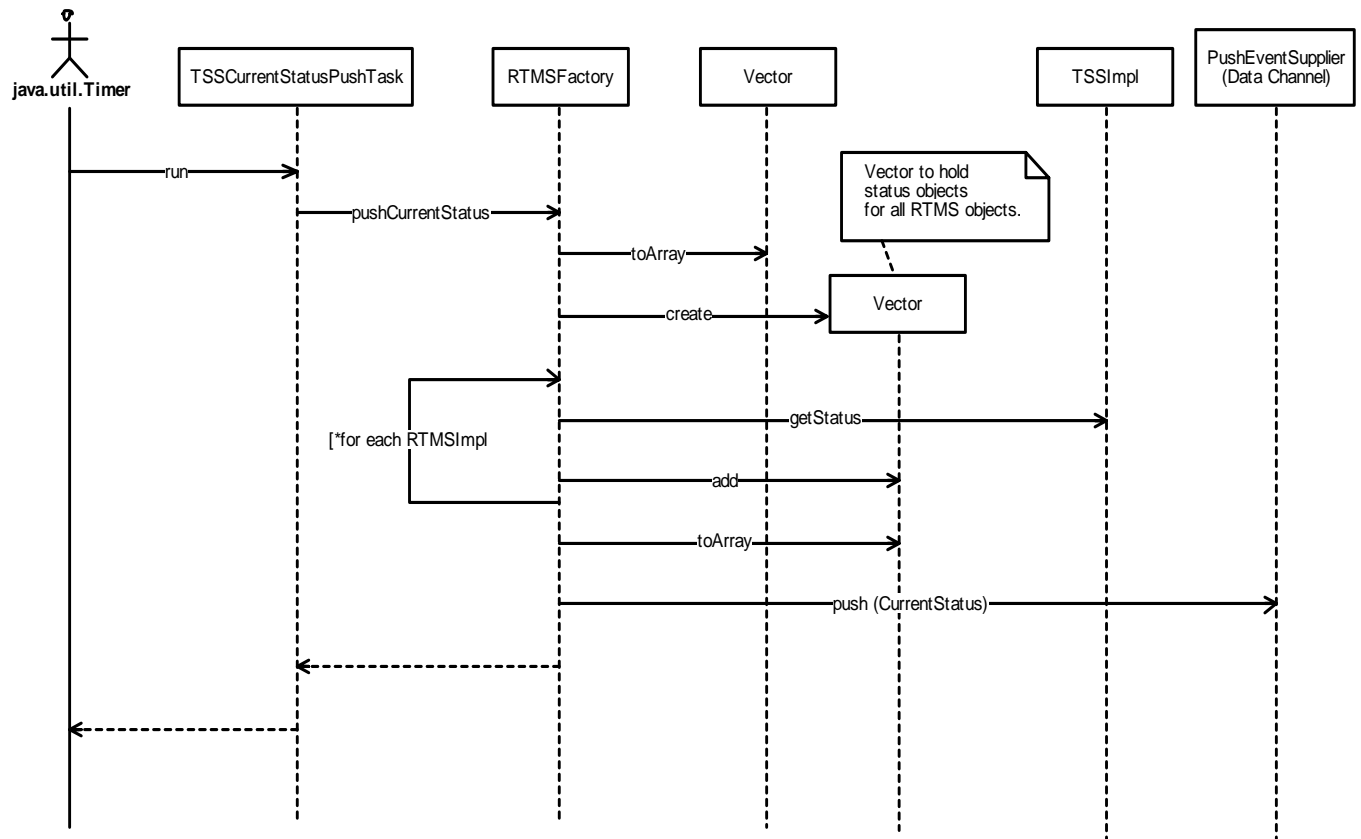


Figure 9-8. RTMSFactoryImpl:CurrentStatusPush (Sequence Diagram)

### 9.3.2.6 RTMSFactoryImpl:pollDevices (Sequence Diagram)

The factory timer runs the PollTSSSTask which executes pollDevices() on the RTMSFactory. pollDevices() gets the list of all RTMSs contain in the factory and operates the following procedure on each. It calls isPollNecessary on each RTMSImpl in a common connection vector. If conditions are met to poll the device a PollTSSCmd is placed on the command queue and later executed. If no CommandQueue exist for the connection group yet one is created. Each command placed on the queue uses the same command queue if its Impl resides in the same vector.

The same vector is then use to check each RTMSImpl in the vector to see if BIT is necessary. If BIT for the device is enabled and the scheduled time for the BIT has passed then BuiltInTestCmd will be placed on the same command queue as poll command and executed when its turn is reached on the command queue. When the command queue is empty the data port is released.

After the function has operated on all Impls in a vector it gets the next vector in the list until all vectors and Impls have been checked for both poll and BIT operation.

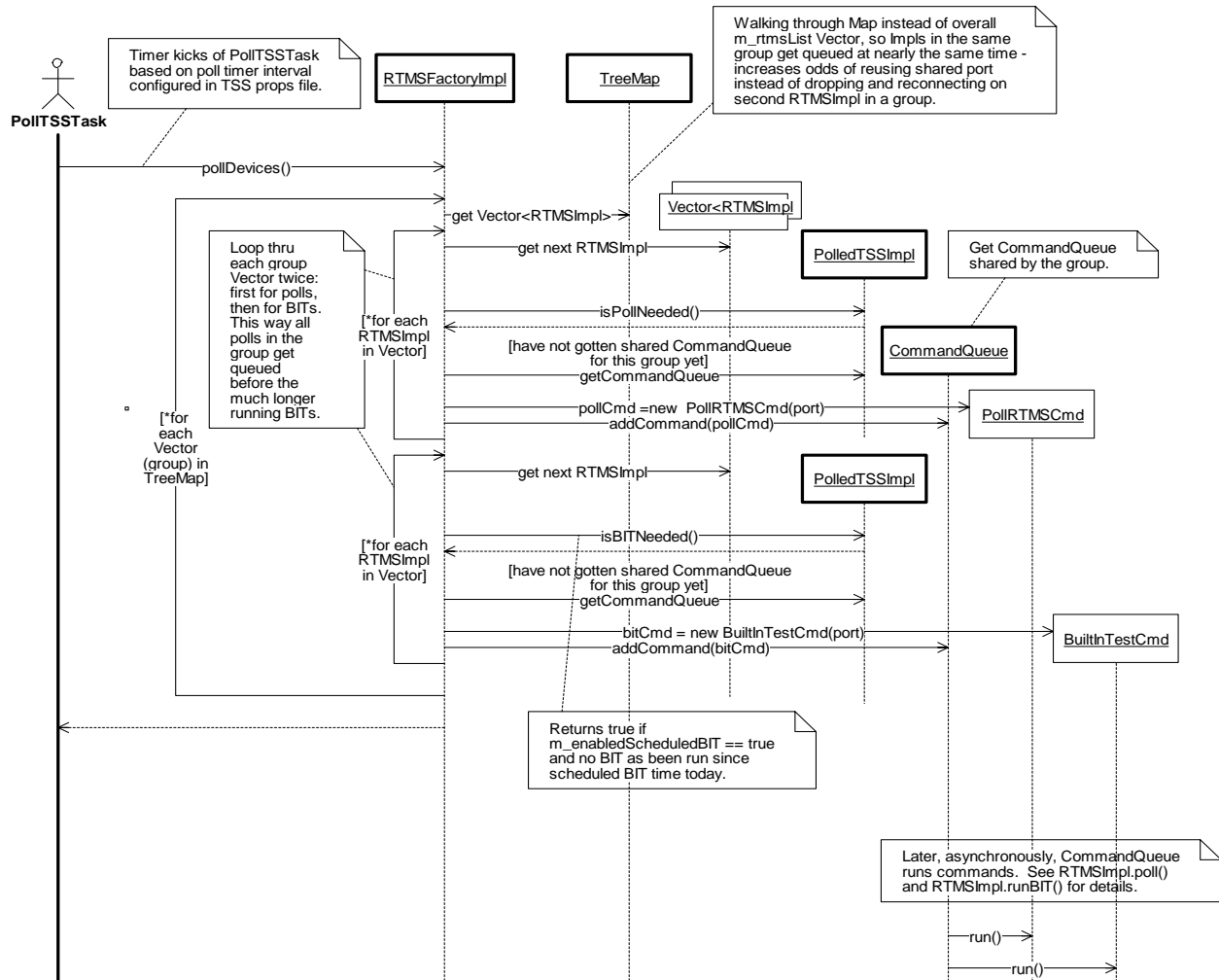


Figure 9-9. RTMSFactoryImpl:pollDevices (Sequence Diagram)

### 9.3.2.7 RTMSImpl:runBIT (Sequence Diagram)

This method is used to execute the built-in test on RTMS. The function checks to see if the user making the request has the right to execute the command. If the user does not have the proper rights an AccessDenied exception is thrown. A new BuiltInTestCmd is created if the user has the needed rights. The command is then added to the command queue. A command status update "Built In Test Command queued" is sent. At some point the builtInTestImpl() is run from the command queue. The BIT last run time is updated. If the built-in test completes the results are saved for later analysis. If the Built-in Test fails the time of failure is saved. If the connection is not made or some other exception occur a command status completed failure is sent and the opStatus is set to COMM\_FAILURE. The COMM status is updated on the database regardless of its value. The BIT results are analysed using the proper bit mask for the RTMS model type. If any hardware fault is returned the device OpStatus is set to HARDWARE\_FAILURE. Both the operation status and hardware status are pushed onto the event queue. After BIT is completed successfully or by exception, the function checks the command queue to see if it is empty. If the queue is empty the port connection is released and the TSSGroup's connected port info is set to null.

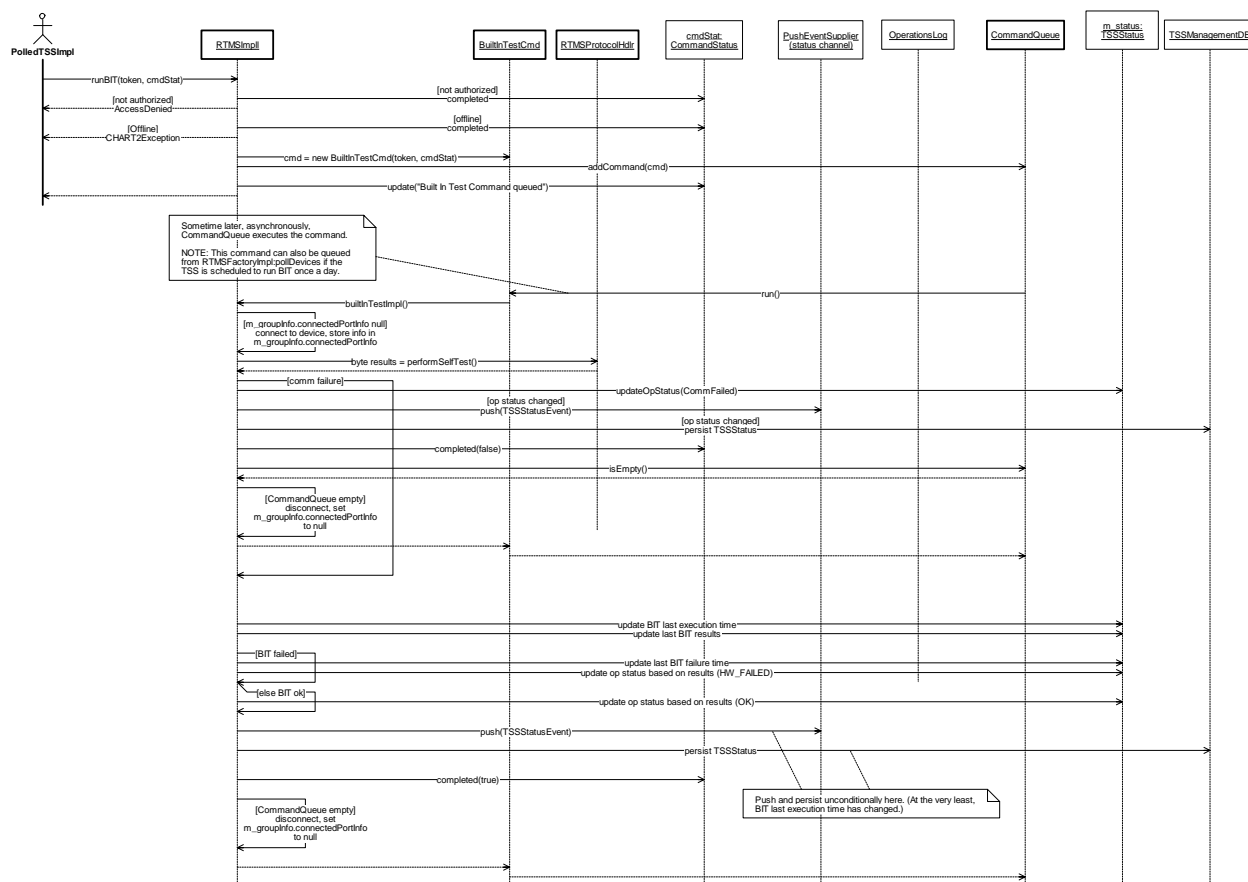


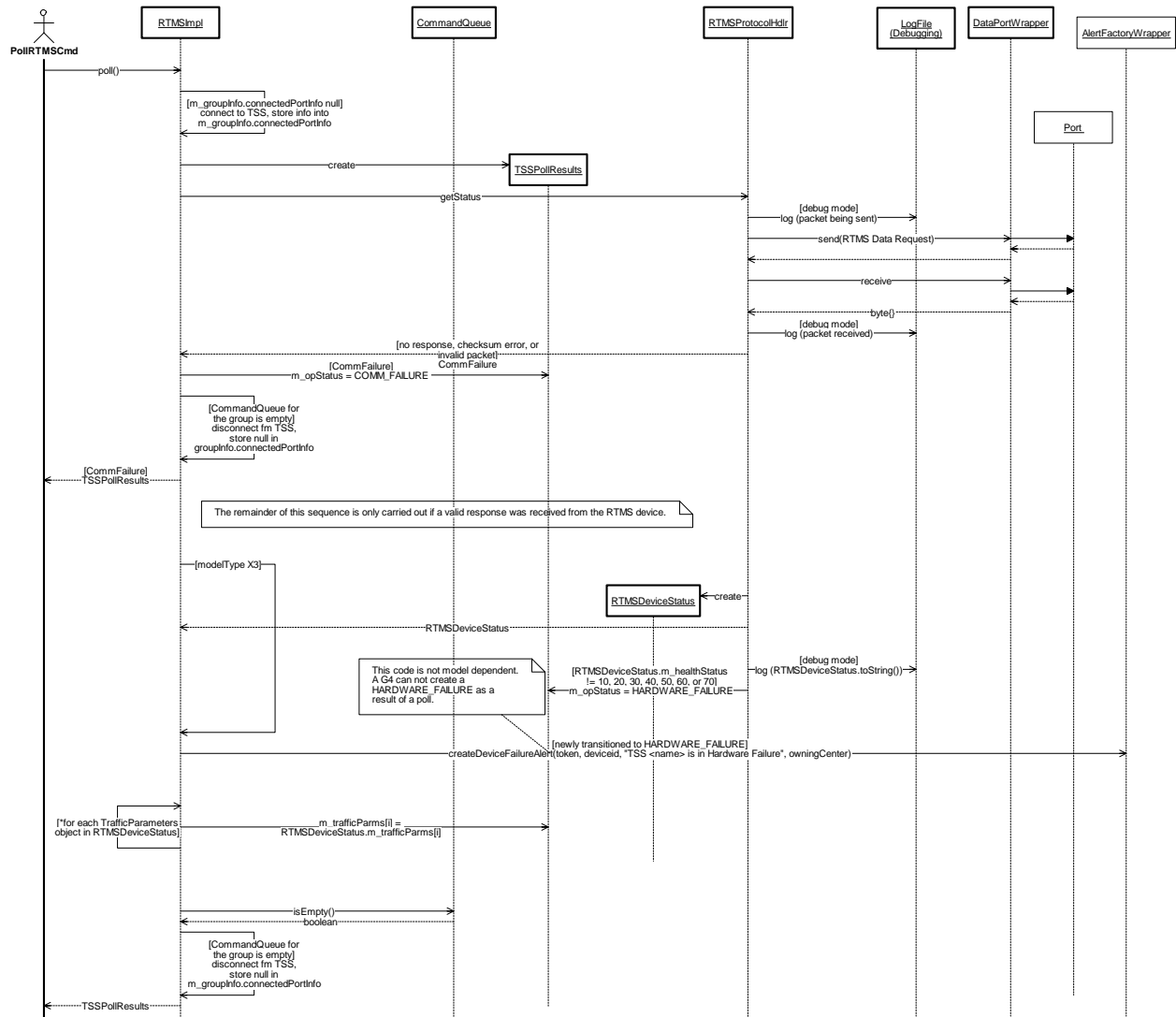
Figure 9-10. RTMSImpl:runBIT (Sequence Diagram)

### 9.3.2.8 RTMSImpl:poll (Sequence Diagram)

The CommandQueue executes the PollRTMSCmd which executes pollImpl() on the RTMSImpl. This function creates a new TSSPollResults objects and executes getStatus() on the protocol handler. The RTMSImpl uses the RTMSProtocolHandler to send a data request to the device and parse the device response. Any communication failure, such as a non-responsive device, causes the base class to be notified that a communication failure occurred. If a communication failure did not occur, the X3 RTMS health status is checked for an indication of a hardware failure. If no hardware failure exists, the lane level data is passed back to the base class to process the data.

A DeviceFailureAlert is created only when the RTMS transitions from another state into HARDWARE\_FAILURE (X3 RTMS only). Any future transitions into another state have no effect on the alert. A device that cycles in and out of a hardware failure causes this class to generate many DeviceFailure alerts however it is left to the AlertModule to not create duplicate open alerts.

When the poll is complete successfully or by exception, the function checks the command queue to see if it is empty. If the queue is empty the port connection is removed and the TSS group info's connected port info is set to null.



**Figure 9-11. RTMSImpl:poll (Sequence Diagram)**



## 9.4 chartlite.data.tss-data

### 9.4.1 Class Diagrams

#### 9.4.1.1 GUITSSDataClasses (Class Diagram)

This diagram shows objects related to adding TCP/IP connection functionality and geo location to TSS's.

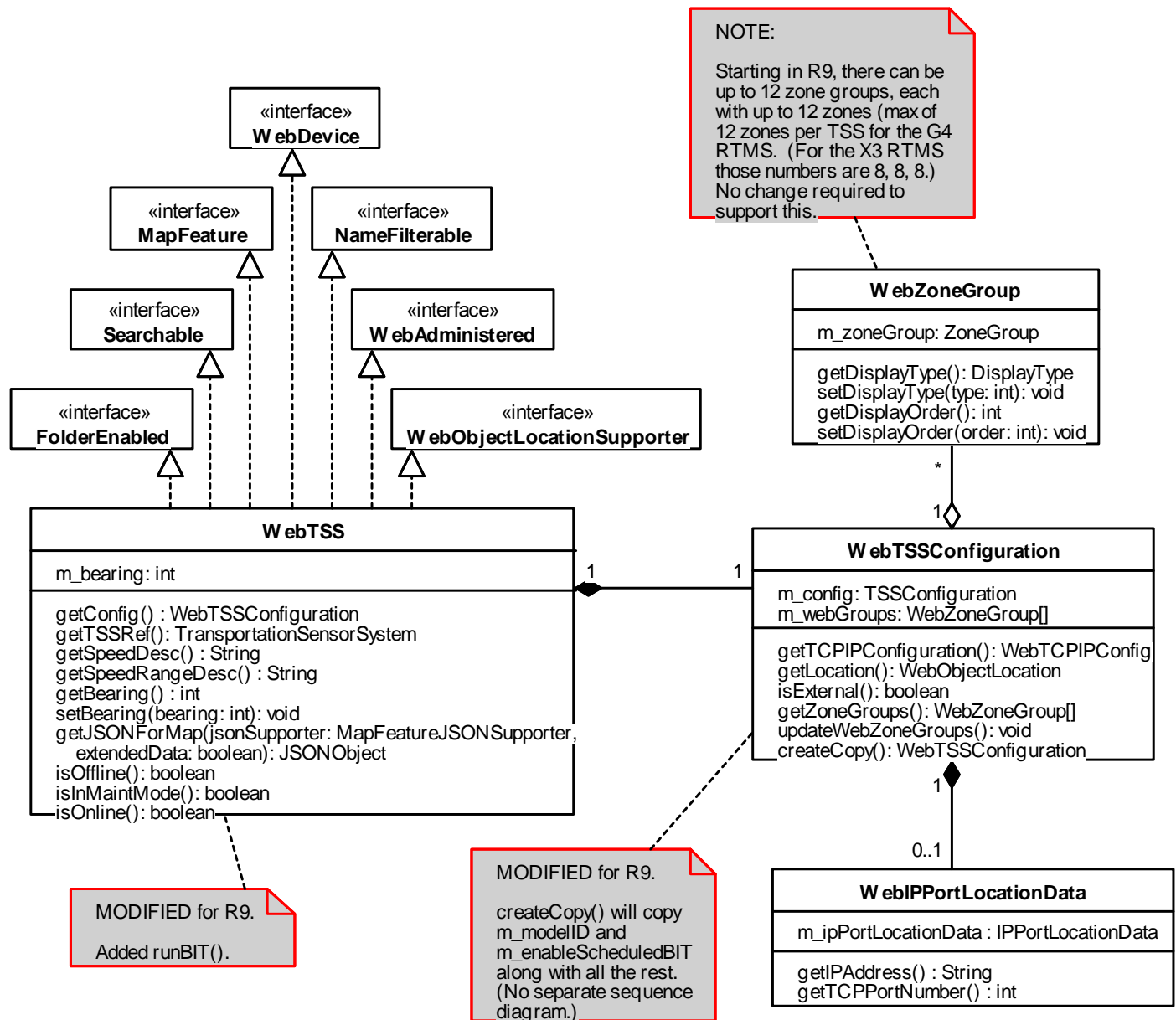


Figure 9-12. GUITSSDataClasses (Class Diagram)

#### **9.4.1.1.1 FolderEnabled (Class)**

This interface provides access to information about an object that can be stored in a folder.

#### **9.4.1.1.2 MapFeature (Class)**

This interface provides data necessary for displaying a feature on a map.

#### **9.4.1.1.3 NameFilterable (Class)**

This java interface is implemented by classes which can be filter by name within the ObjectCache. A NameFilter object is passed into the ObjectCache to select NameFilterable objects in the cache.

#### **9.4.1.1.4 Searchable (Class)**

This interface allows objects to be searched for via a substring search.

#### **9.4.1.1.5 WebAdministered (Class)**

This interface allows the implementing class to be administered via the trader console pages.

#### **9.4.1.1.6 WebDevice (Class)**

This interface contains common functionality for CHART devices.

#### **9.4.1.1.7 WebIPPortLocationData (Class)**

This class wraps the IPPortLocationData IDL structure and provides accessor methods to get the data. This class has data for identifying a TCP/IP address and port.

#### **9.4.1.1.8 WebObjectLocationSupporter (Class)**

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

#### **9.4.1.1.9 WebTSS (Class)**

This class wraps the TransportationSystemSensor CORBA interface, caches data, and provides access to the cached data.

#### **9.4.1.1.10 WebTSSConfiguration (Class)**

This class wraps the TSSConfiguration IDL structure and provides accessors for easy access to the data.

#### **9.4.1.1.11 WebZoneGroup (Class)**

This class wraps the ZoneGroup IDL structure and provides accessors for easy access to the data.

## 9.5 chartlite.servlet.tss

### 9.5.1 Class Diagrams

#### 9.5.1.1 chartlite.servlet.tss\_classes (Class Diagram)

This diagram shows CHART GUI servlet classes related to traffic sensor signs.

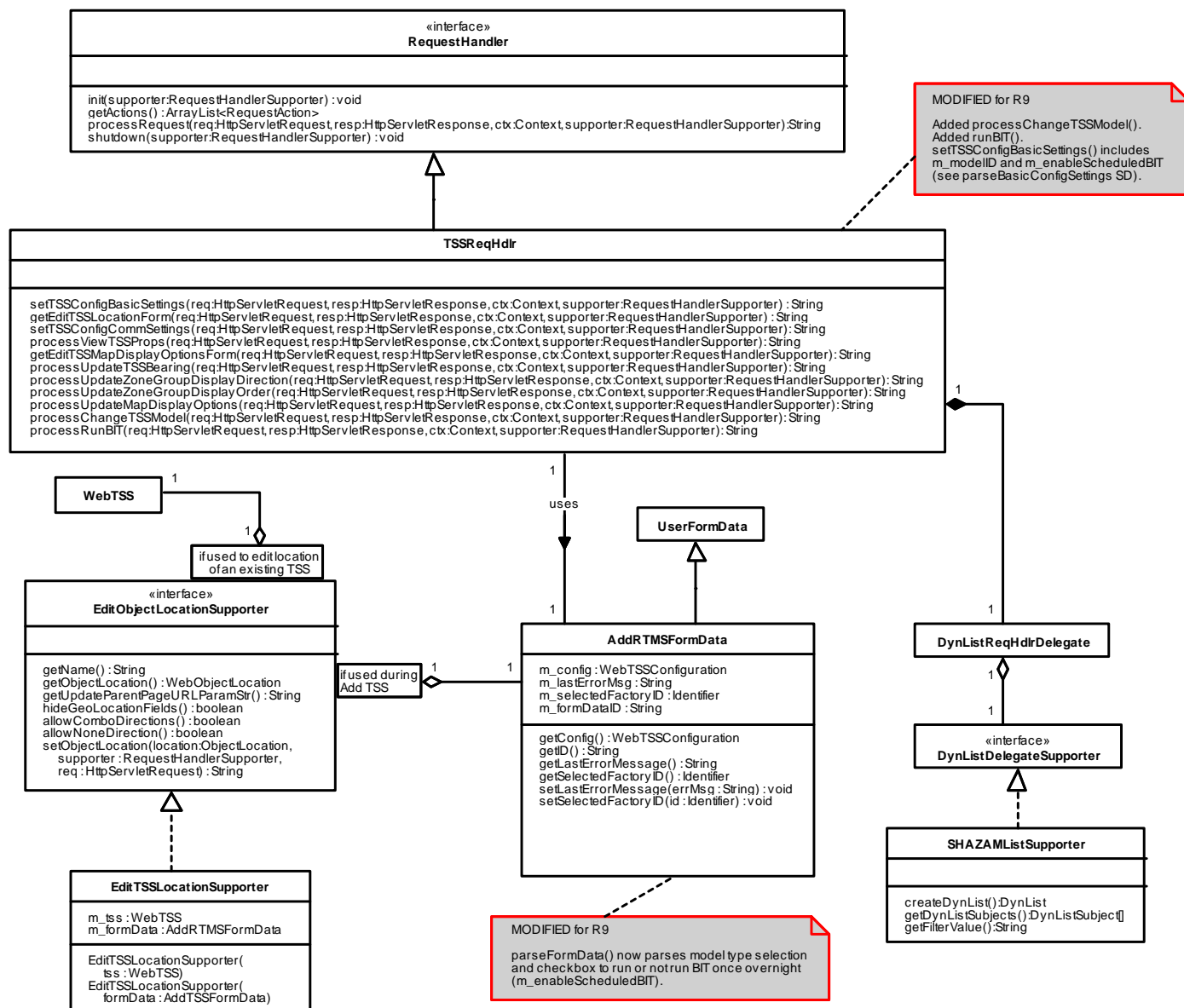


Figure 9-13. chartlite.servlet.tss\_classes (Class Diagram)

#### **9.5.1.1.1 AddRTMSFormData (Class)**

This class represents the data in the Add RTMS form.

#### **9.5.1.1.2 DynListDelegateSupporter (Class)**

This interface contains functionality to support the DynListReqHdlrDelegate

#### **9.5.1.1.3 DynListReqHdlrDelegate (Class)**

This class helps request handlers support dynamic lists. Requests to view, sort, or filter dynamic lists can be passed from a request handler to this class, provided the URL used for the requests contain parameters required by this class, such as the id of the list, the property name, and/or the filter value.

#### **9.5.1.1.4 EditObjectLocationSupporter (Class)**

This interface provides functionality allowing the location data to be edited. (For example, the target of the edited location may be an existing object, or it may be a form data object for creating a new object).

#### **9.5.1.1.5 EditTSSLocationSupporter (Class)**

This class is used to support editing the location of an existing or new TSS.

#### **9.5.1.1.6 RequestHandler (Class)**

This interface specifies methods that are to be implemented by classes that are used to process requests.

#### **9.5.1.1.7 SHAZAMListSupporter (Class)**

This class is a DynListDelegateSupporter for the SHAZAM dynamic list. It provides SHAZAM specific functionality to the generic DynListReqHdlrDelegate.

#### **9.5.1.1.8 TSSReqHdlr (Class)**

This class handles requests related to traffic sensor systems such as RTMS.

#### **9.5.1.1.9 UserFormData (Class)**

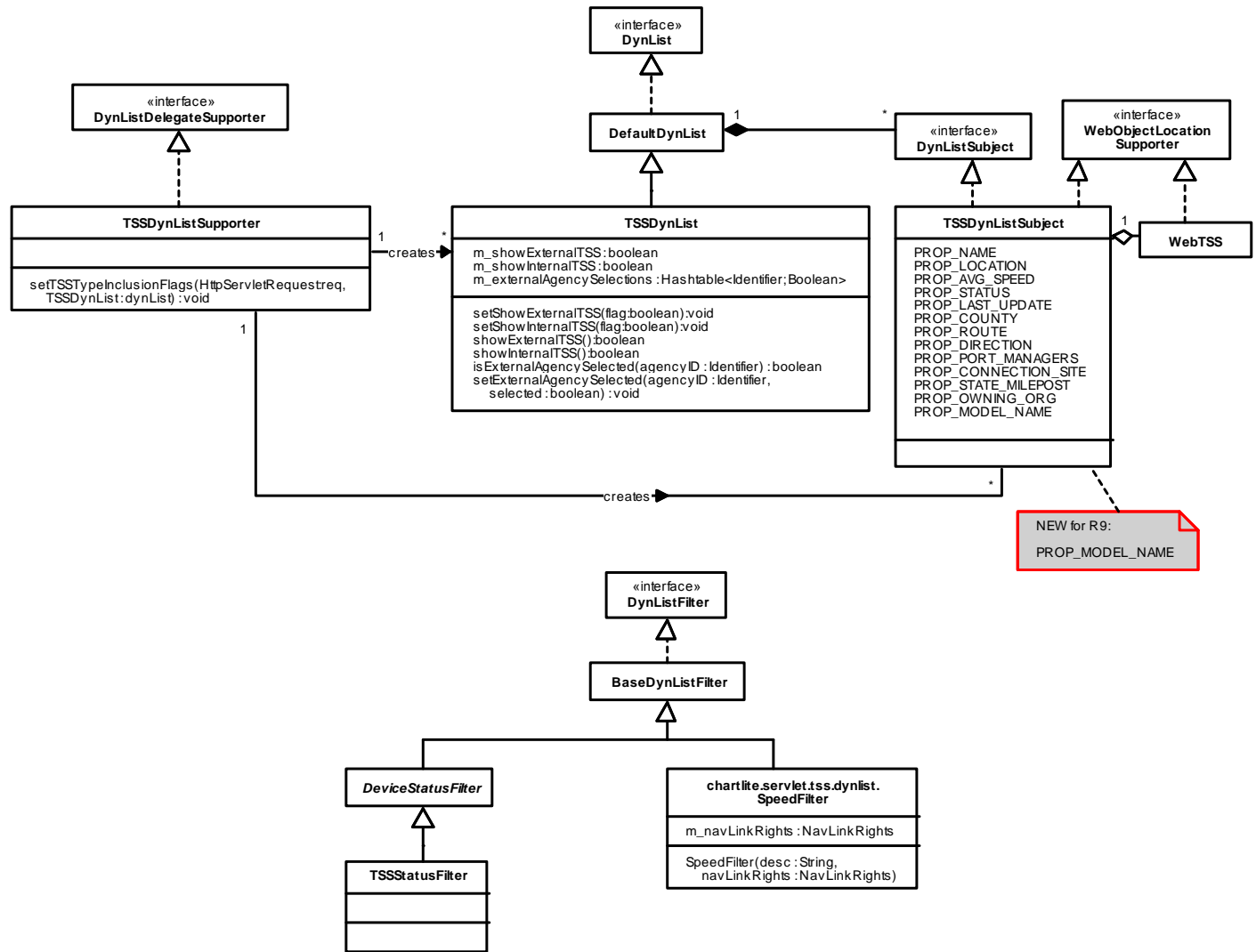
This class is used to store form data between requests while a user is editing a complex form, and provides convenience methods for parsing the values from the request.

#### **9.5.1.1.10 WebTSS (Class)**

This class wraps the TransportationSystemSensor CORBA interface, caches data, and provides access to the cached data.

### 9.5.1.2 chartlite.servlet.tss\_dynlist\_classes (Class Diagram)

This diagram shows classes related to using TSS devices in dynamic lists.



**Figure 9-14. chartlite.servlet.tss\_dynlist\_classes (Class Diagram)**

#### 9.5.1.2.1 BaseDynListFilter (Class)

This abstract class provides a base implementation of the DynListFilter interface.

#### 9.5.1.2.2 chartlite.servlet.tss\_dynlist.SpeedFilter (Class)

This class is a filter for the TSS dynamic list that filters on the current speed value. For R3B3, it will examine the user's rights for each TSS to determine whether the user is allowed to view the average speed or speed ranges for a TSS, and will apply the filter based on the viewable data.

#### **9.5.1.2.3 DefaultDynList (Class)**

This class provides a default implementation of the DynList interface. It supports a collection of columns, a collection of global filters, and a collection of subjects. Filters in this list are treated additively - that is, a subject must pass all filters to be displayed.

#### **9.5.1.2.4 DeviceStatusFilter (Class)**

This class is a base filter for filtering on the device status. It must be extended to get the WebDevice from the DynListSubject object.

#### **9.5.1.2.5 DynList (Class)**

This interface is implemented by classes that wish to provide dynamic list capabilities. A dynamic list is a list of items that has one or more columns that can optionally be sorted, and the list can be filtered by column values or by global filters.

#### **9.5.1.2.6 DynListDelegateSupporter (Class)**

This interface contains functionality to support the DynListReqHdlrDelegate

#### **9.5.1.2.7 DynListFilter (Class)**

This interface is implemented by classes that are used to filter dynamic lists.

#### **9.5.1.2.8 DynListSubject (Class)**

This interface is implemented by classes that wish to be capable of being displayed in a dynamic list.

#### **9.5.1.2.9 TSSDynList (Class)**

This class implements the dynlist interface for TSS's in dynamic lists.

#### **9.5.1.2.10 TSSDynListSubject (Class)**

This class implements the DynListSubject interface and contains fields for sorting and filtering TSS's in dynamic lists.

#### **9.5.1.2.11 TSSDynListSupporter (Class)**

This class implements the DynListDelegateSupporter for creating dynamic lists containing TSS's

#### **9.5.1.2.12 TSSStatusFilter (Class)**

This filter extends the DeviceStatusFilter to provide status filtering for TSS objects.

#### **9.5.1.2.13 WebObjectLocation Supporter (Class)**

This interface allows common processing for objects supporting an ObjectLocation via the WebObjectLocation wrapper class.

#### **WebTSS (Class)**

This class wraps the TransportationSystemSensor CORBA interface, caches data, and provides access to the cached data.

## 9.5.2 Sequence Diagrams

### 9.5.2.1 chartlite.servlet.tss:parseBasicConfigSettings (Sequence Diagram)

This diagram shows the processing done by the TSSReqHdlr to parse the parameters passed from the TSS basic configuration data form. This existing method is being modified for R3B1 to allow the responsible operations center to be set. This optional field will be used to specify which operations center should receive a device failure alert for a TSS. When not set, device failure alerts for the TSS are disabled.

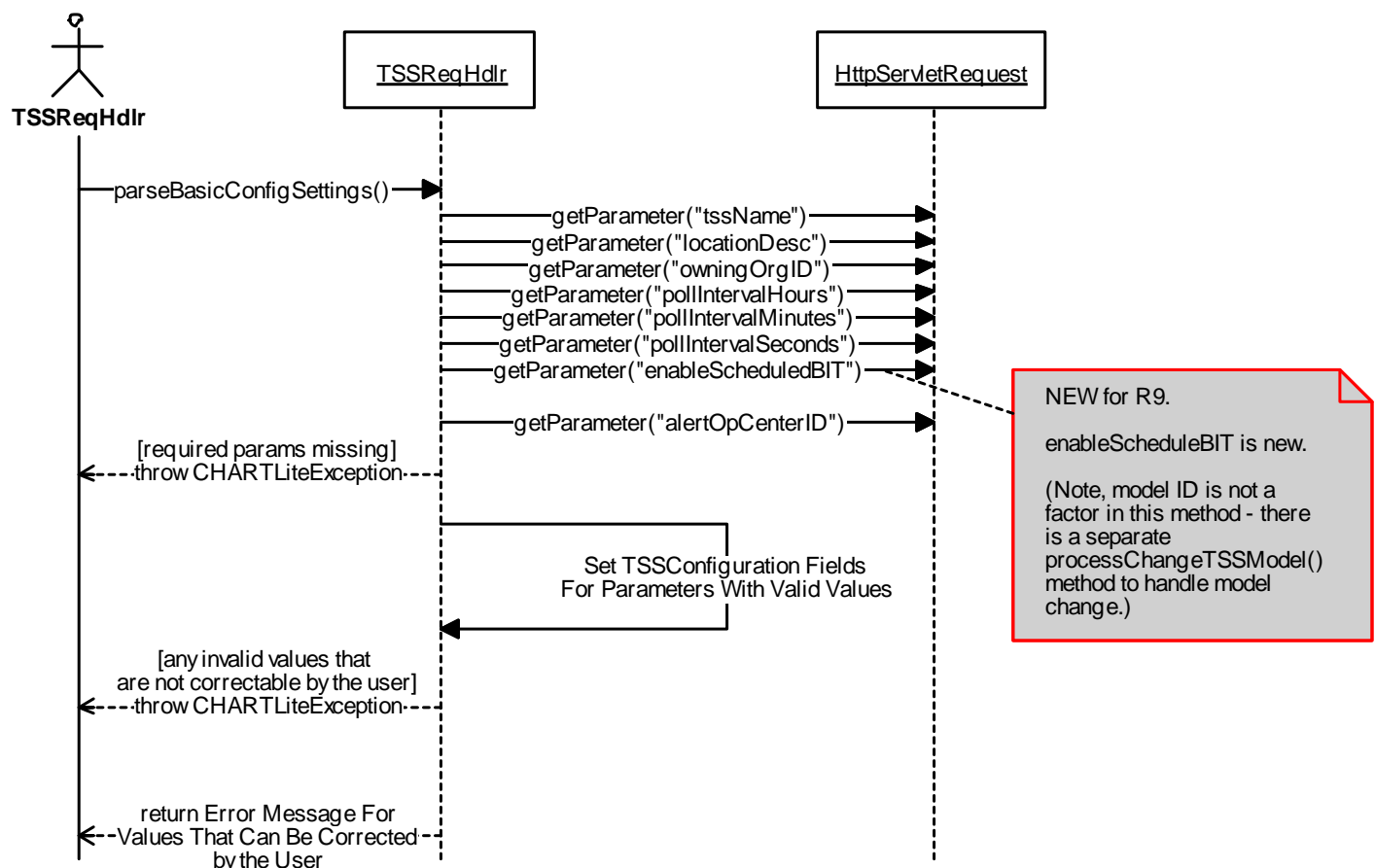
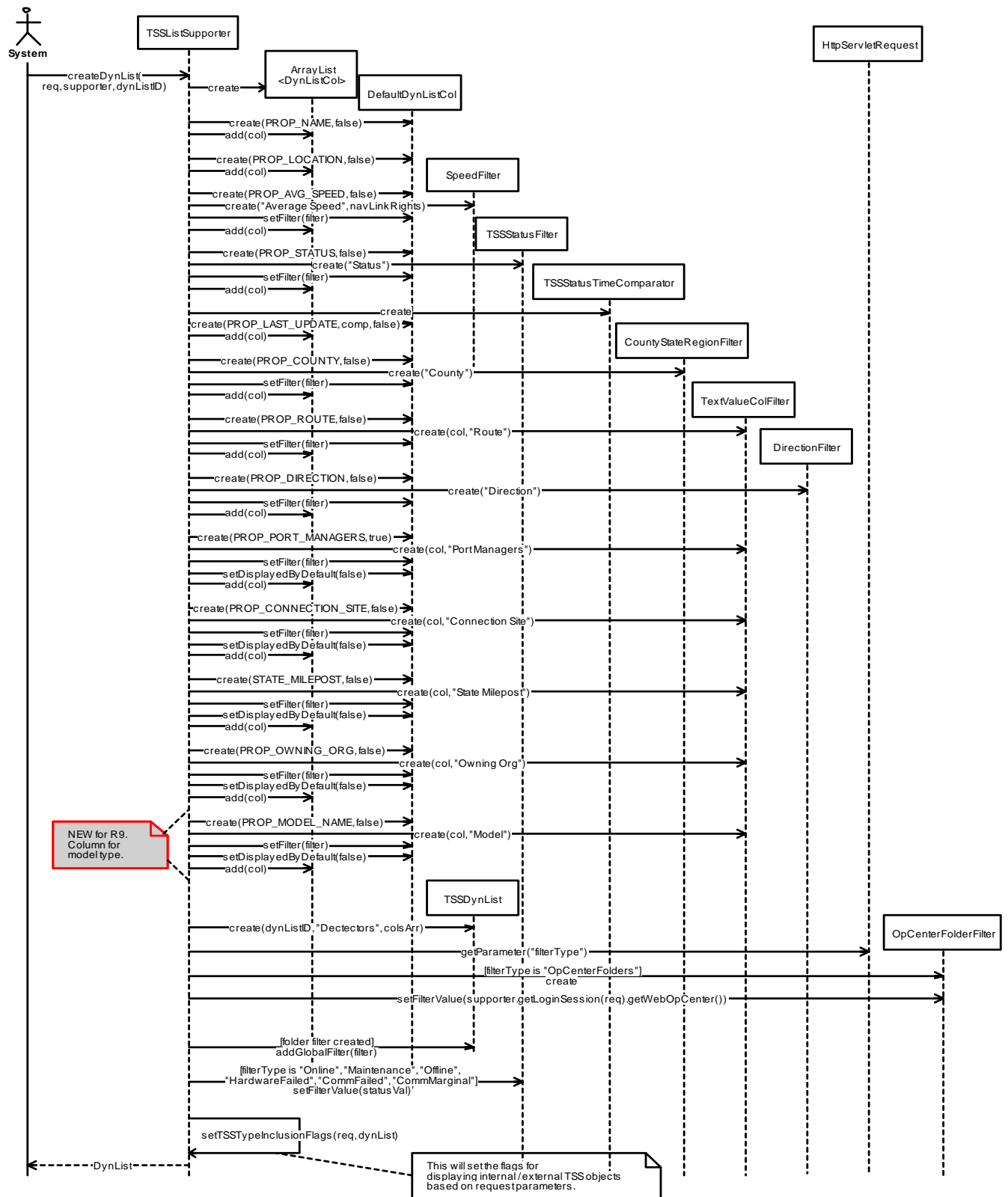


Figure 9-15. chartlite.servlet.tss:parseBasicConfigSettings (Sequence Diagram)

### 9.5.2.2 TSSListSupporter:createDynList (Sequence Diagram)

This diagram shows how the Detector/TSS dynamic list is created for viewing on the Detector List page. A DefaultDynListCol object is created to represent each column. If a column has a filter, the filter object is created and set into the column. If the column is to be hidden by default, a call is made to the column set the default visibility. A new TSSDynList object is created using the columns that were created. If the request indicates that a global filter for the user's operation center folders should be used, the filter is created and added to the dynamic list. Other filter values include one of the device status values, which (if specified) is set in the device status filter. The setTSSTypeInclusionFlags() method is called to apply the external / internal TSS flags specified in the request.





**Figure 9-16. TSSListSupporter:createDynList (Sequence Diagram)**

### 9.5.2.3 TSSReqHdlr:processAddTSS (Sequence Diagram)

This diagram shows the processing that is performed when the administrator chooses to add a TSS to the system or create a new TSS by copying an existing TSS. This processing is also performed if an add or copy is already in progress and the Add form is being redisplayed after navigating away from the add form to set the location data. If this is the case, a formID parameter will be used to retrieve the form data object from the temp object store. If performing a copy, the WebTSS being copied will be retrieved from the object cache and will be used to create an AddTSSFormData object. This will have the effect of pre-populating the Add TSS form with data from the TSS being copied. If this is an Add TSS operation that was not already in progress (neither formID nor TSSID is present in the request), a new empty AddTSSFormData object is created. The form data is stored in the temp object store if not already stored there. Other objects needed for the form for select lists are obtained from the object cache and placed in the context. The Add TSS form is then displayed to the user.

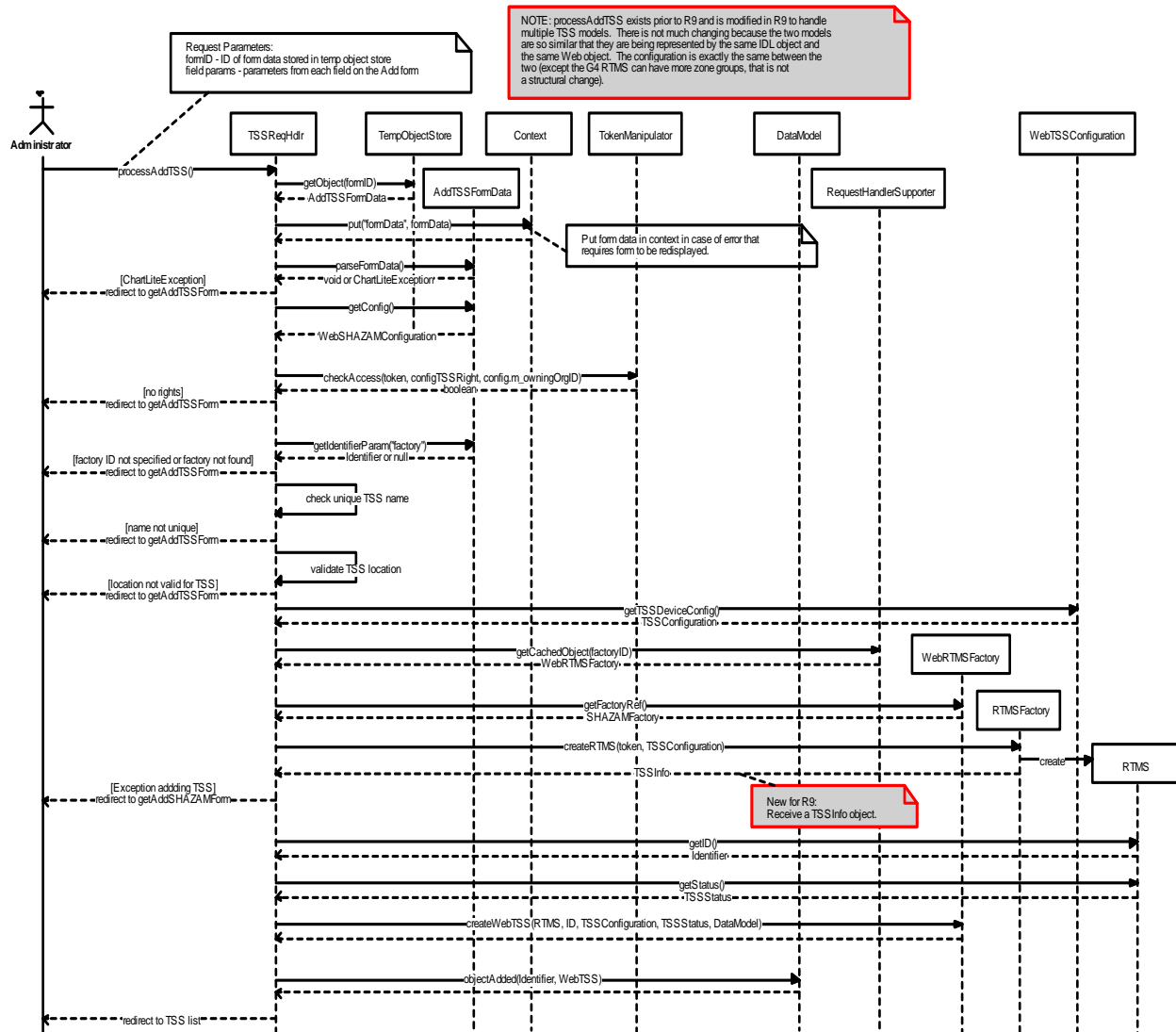


Figure 9-17. TSSReqHdlr:processAddTSS (Sequence Diagram)

#### 9.5.2.4 TSSReqHdlr:processChangeTSSModel (Sequence Diagram)

This diagram shows the processing that is performed when the user submits the form used to change the TSS model. The parameters are read and the WebTSS object for the specified TSS is retrieved from the data model. The user's rights are checked to make sure the user has permission to modify the configuration for the TSS. The TSS status is checked to make sure the TSS is offline. The TransportationSensorSystem CORBA object reference is obtained and a CommandStatus object is created. The TransportationSensorSystem CORBA object is called to change the model; the status of the operation will be reported back to the user via the CommandStatus object. The user's browser is then redirected to the command status page where the command progress can be monitored.

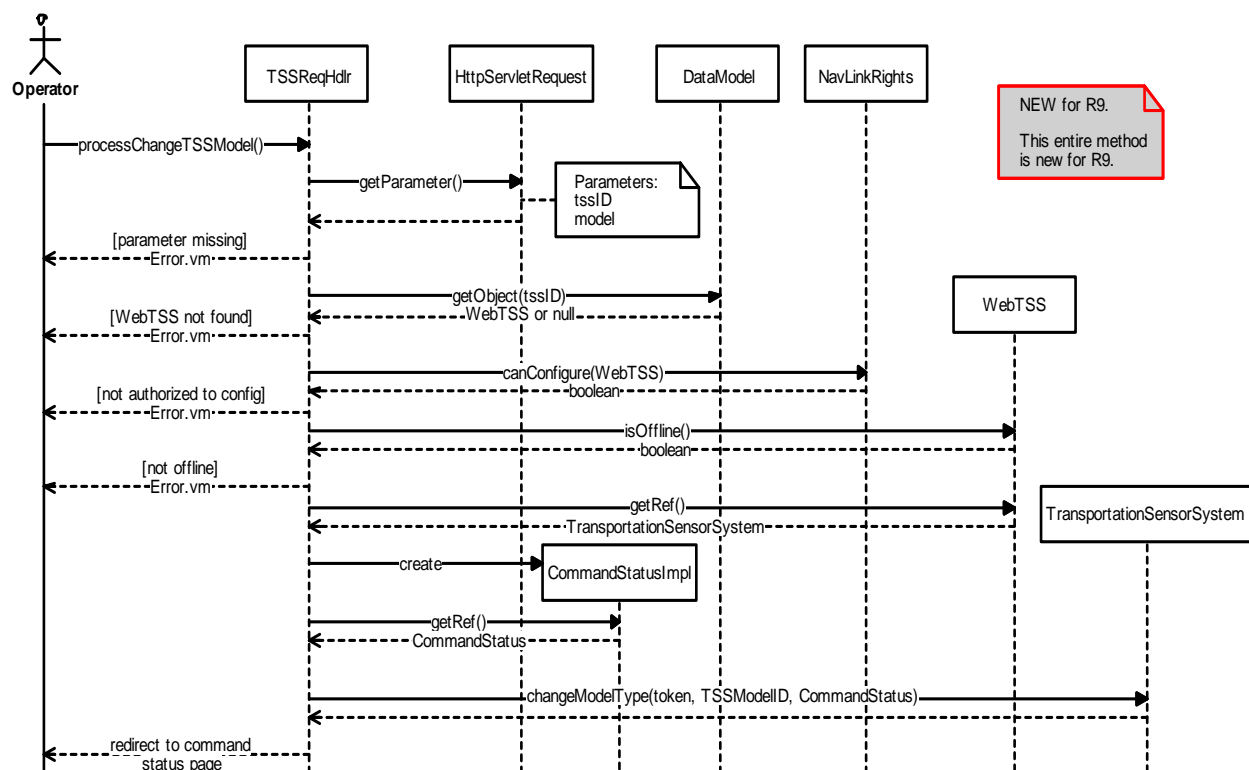


Figure 9-18. TSSReqHdlr:processChangeTSSModel (Sequence Diagram)

### 9.5.2.5 TSSReqHdlr:processRunBIT (Sequence Diagram)

This diagram shows the processing that occurs when the user chooses to run Built-in Test on a TSS. The id of the TSS is obtained from the request and the ID is used to find the TSS in the DataModel. The TSS must be in maintenance mode and the user must have the maintainTSS right for the TSS's owning organization or an error is displayed. A command status object is created and passed along with the user's token in a request to the CORBA TSS object to run BIT. If an exception is thrown, the GUI must complete the CommandStatus, otherwise the server will provide updates as it processes the request asynchronously. The request is then redirected to the CommandStatus page where the user can monitor the status of the command.

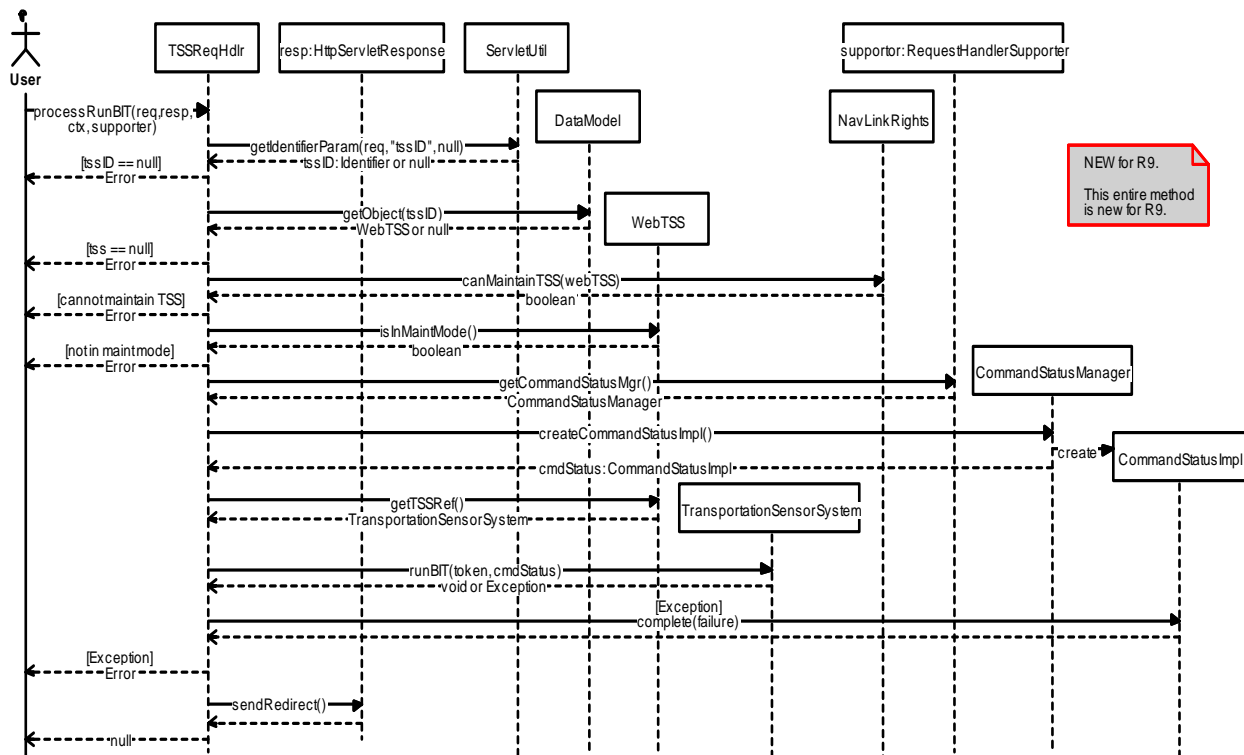


Figure 9-19. TSSReqHdlr:processRunBIT (Sequence Diagram)

## **10 Deprecated Functionalities**

---

No CHART functionality is being deprecated with the introduction of R9.

## 11 Mapping To Requirements

The following table shows how the requirements in the CHART R9 Requirements document map to design elements contained in this design.

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.1.1.4.10.1	The system shall allow the user to specify the total number of simultaneous video sessions that will be allowed for all users of the Center, when adding a new Center.	Video	Configure Operations Center	ResourceManagement CD
SR1.1.1.4.11.2	The system shall display the maximum number of desktop video sessions that all users of the operations center (combined) are allowed to display.	Video	View Operations Center List	MiscDataClasses CD
SR1.1.1.4.11.3	The system shall display the total number of active desktop video sessions that are allocated to all users logged into the operations center.	Video	View Operations Center List	MiscDataClasses CD
SR1.1.1.4.14.4	The system shall allow the user to modify the total number of simultaneous video sessions that will be allowed for all users of an existing Center.	Video	Configure Operations Center	ResourceManagement CD
SR1.5.2.1.5.4	The system shall support TSS communication via a Field Management Server that implements RS232, ISDN, POTS, and direct-connect communications mediums.	TSS	Set Modem Communication Settings Set Direct Communication Settings Poll TSS	(existing fcn now incl G4)
SR1.5.2.1.5.7	The system shall support TSS	TSS	Set TCP/IP	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	communication via a TCP/IP communications medium.		Communication Settings Poll TSS	
SR1.5.2.1.17.10	Add/Copy TSS	TSS	Add TSS	
SR1.5.2.1.17.10.2	The system shall allow the user to specify the configuration parameters of the new TSS, as defined by the Specify TSS Configuration requirements.	TSS	Set TSS Configuration Specify TSS Model	TSSReqHdr:processAddTSS SD
SR1.5.2.1.17.10.4	The system shall allow the user to pre-populate the configuration settings for creating a new TSS using the settings of an existing TSS.	TSS	Copy TSS	TSSReqHdr:processAddTSS SD
SR1.5.2.1.17.11	Configure TSS	TSS	Set TSS Configuration	
SR1.5.2.1.17.11.1	The system shall allow a suitably privileged user to change the configuration parameters for a TSS in maintenance mode or offline mode.	TSS	Set TSS Configuration	(existing fcn now incl G4)
SR1.5.2.1.17.11.2	The system shall allow the user to specify the configuration parameters for the existing TSS, as defined by the Specify TSS Configuration requirements.	TSS	Set TSS Configuration	(existing fcn now incl G4)
SR1.5.2.1.17.12	Remove TSS	TSS	Remove TSS	
SR1.5.2.1.17.12.1	The system shall allow a suitably privileged user to remove a TSS from the system.	TSS	Remove TSS	(existing fcn now incl G4)
SR1.5.2.1.17.12.2	The system shall support the Remove TSS command only if the TSS is in offline mode.	TSS	Remove TSS	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR1.5.2.1.17.1 2.3	The system shall prompt the user for confirmation before removing a TSS.	TSS	Remove TSS	(existing fcn now incl G4)
SR1.5.2.1.17.1 3	Specify TSS Configuration	TSS	Set TSS Configuration	
SR1.5.2.1.17.1 3.4	The system shall allow the user to specify the polling interval for the TSS.	TSS	Set TSS Configuration	(existing fcn now incl G4)
SR1.5.2.1.17.1 3.4.1	The minimum TSS polling interval shall be 1 second.	TSS	Set TSS Configuration Poll TSS	(existing fcn re-impl for G4)
SR1.5.2.1.17.1 3.4.2	The maximum TSS polling interval shall be 24 hours.	TSS	Set TSS Configuration Poll TSS	(existing fcn re-impl for G4)
SR1.5.2.1.17.1 3.4.3	The resolution for the TSS polling interval shall be in units of 1 second.	TSS	Set TSS Configuration	(existing fcn now incl G4)
SR1.5.2.1.17.1 3.5	The system shall allow the user to specify whether or not to enable the device communication log for the TSS.	TSS	Set TSS Configuration Poll TSS	(existing fcn now incl G4)
SR1.5.2.1.17.1 3.6	The system shall allow the user to specify the grouping of the TSS detection zones into one or more logical zone groups.	TSS	Set TSS Configuration Poll TSS	(existing fcn now incl G4)
SR1.5.2.1.17.1 3.6.5	A zone group shall include a set of zones.	TSS	Set TSS Configuration Poll TSS	(existing fcn now incl G4)
SR1.5.2.1.17.1 3.17	The system shall allow the user to specify the model of the TSS.	TSS	Change TSS Model Specify TSS Model	TSSReqHdlr:processChangeTSSModel SD PolledTSSImpl:changeModelType SD TSSReqHdlr:processAddTSS SD RTMSFactoryImpl:createRTMS SD
SR1.5.2.1.17.1 3.17.1	The system shall support the ISS X3 RTMS model of TSS.	TSS	Change TSS Model Specify TSS Model	PolledTSSImpl:changeModelType SD TSSReqHdlr:processChangeTSSModel SD
SR1.5.2.1.17.1 3.17.1.1	The system shall allow the user to specify up to 8 zones (lanes) for the ISS X3 RTMS when configuring zone groups as defined under	TSS	Configure TSS Zone Groups	PolledTSSImpl:changeModelType SD TSSReqHdlr:processChangeTSSModel SD



Tag	Requirement	Feature	Use Cases	Other Design Elements
	requirement SR1.5.2.1.17.13.6.			
SR1.5.2.1.17.1 3.17.1.1.1	If a user changes the model type of a G4 RTMS to an X3 RTMS, the system shall delete any zones beyond the first 8, and any zone groups which are left empty by this action. (See also requirement SR1.5.2.1.17.13.6 covering configuration of zone groups.)	TSS	Change TSS Model	PolledTSSImpl:changeModelType.etd
SR1.5.2.1.17.1 3.17.2	The system shall support the ISS G4 RTMS model of TSS.	TSS	Change TSS Model Specify TSS Model	PolledTSSImpl:changeModelType.etd
SR1.5.2.1.17.1 3.17.2.1	The system shall allow the user to specify up to 12 zones (lanes) for the ISS G4 RTMS when configuring zone groups as defined under requirement SR1.5.2.1.17.13.6.	TSS	Configure TSS Zone Groups	PolledTSSImpl:changeModelType SD
SR1.5.2.1.17.1 3.18	The system shall allow the user to specify whether CHART is to automatically command this particular TSS to run its Built-in Test (BIT) every day at or soon after the TSS BIT time of day specified in the System Profile.	TSS	Set TSS Configuration	RTMSImpl:executeBIT SD
SR1.5.2.1.18.8 .12.4	The system shall allow the user to specify whether the Streaming Flash Server is classified as "public" for the purposes of blocking / unblocking the given camera's display to the public.	Video	Configure Flash Streaming Control	none (prototype only)
SR1.5.2.1.18.1 6.5	The system shall allow the user to specify whether the Streaming Flash Server is to be classified as "public" for the purposes of blocking /	Video	Configure Flash Server	none (prototype only)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	unblocking camera displays to the public.			
SR1.5.3.1.7.1	The system shall allow a user with the Maintain TSS right to command an RTMS in maintenance mode to execute a Built-in Test (BIT).	TSS	Run TSS BIT	TSSReqHdr:processRunBIT SD RTMSImpl:executeBIT SD
SR1.5.4.4.1	The system shall issue a Device Failure Alert when the detector reports a hardware failure, if a responsible Center is defined for that detector.	TSS	Create Alert	(existing fcn now incl G4)
SR1.5.4.4.2	The system shall automatically command an internal online RTMS to execute a Built-in Test (BIT), each day, at or soon after a system-wide administrator-configurable Scheduled BIT Time, if and only if the RTMS is configured to enable scheduled BIT to run.	TSS	Poll TSS	RTMSImpl:executeBIT.etd RTMSFactoryImpl:pollDevices.etd
SR1.5.4.4.2.1	The system shall allow a user with the Configure System right to configure the time of day that CHART commands RTMSs execute their daily Built-in Test (BIT). (This applies only to reachable online internal RTMS models configured to run scheduled BIT.)	TSS	Edit TSS System Profile	
SR1.5.5.4.1.3	The detailed data displayed for a Detector shall include the average speed or average speed range for each zone group, if the user has rights to view the average speed or average speed range for the	TSS	View TSS List	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	detector.			
SR1.5.5.4.1.4	The detailed data displayed for a Detector shall include the Current Status.	TSS	View TSS List	(existing fcn now incl G4)
SR1.5.5.4.1.5	The detailed data displayed for a Detector shall include the Last Update time.	TSS	View TSS List	(existing fcn now incl G4)
SR1.5.5.4.1.13	The system shall allow the user to view additional details for each detector shown in the Detector list. (This is implemented by supplying a link to the TSS Details Page.)	TSS	View TSS List	(existing fcn now incl G4)
SR1.5.5.4.1.14	The detailed data displayed for a Detector shall include the Model Type of the detector.	TSS	View TSS List Select List Columns	
SR1.5.5.4.1.14.1	The Model Type for a Detector within the list shall be hidden by default.	TSS	View TSS List Select List Columns	
SR1.5.5.4.2.14	The system shall allow the user to sort the list of Detectors by Model Type.	TSS	Sort List	
SR1.5.5.4.3.12	The system shall allow the user to filter the list of Detectors by Model Type.	TSS	Filter List	
SR1.5.5.5.1.3.1	The system shall indicate in the local displays data if the camera is displayed in a video session on the user's desktop.	Video	View Video Source List	ViewVideoSourceReqHdlr : processViewVideoSourceListReq SD, ServletBaseClasses CD, MiscDataClasses CD
SR1.5.7.3.17	The system shall show all desktop video sessions displaying the camera.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.17.1	The system shall show the name of	Video	View Video Source	ViewVideoSourceReqHdlr:

Tag	Requirement	Feature	Use Cases	Other Design Elements
	the user owning the desktop video session.		Details	processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.17.2	The system shall show the operations center at which the user owning the desktop video session is logged in.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.17.3	The system shall show the host name and/or IP address of the browser displaying the desktop video session.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.18	The system shall show the status for each Streaming Flash Server configured for a camera.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.18.1	The system shall show the name or IP address of the Streaming Flash Server.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.18.2	The system shall show whether the Streaming Flash Server is designated as "public" or not for the purposes of blocking / unblocking the camera's display to the public.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.3.18.3	The system shall show whether the camera is currently blocked for the Streaming Flash Server or not.	Video	View Video Source Details	ViewVideoSourceReqHdlr: processViewDetailsReq SD; GUIVideoDataClasses CD, GUIVideoServletClasses2 CD
SR1.5.7.5.2	The system shall allow a suitably privileged user to view the configuration information of a TSS in the system.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.1	The system shall allow a user with	TSS	View TSS Details	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	the View TSS Configuration right to view configuration information of an internal TSS.			
SR1.5.7.5.2.2	The system shall allow a user with the View TSS Configuration and View External TSS rights to view configuration information of an external TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.3	The system shall allow a user with the View TSS Configuration right to view the model type of an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.4	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the name of the TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.5	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the network connection site of the TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.6	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the owning organization of the TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.7	The system shall allow a user with the View TSS Configuration right to view the maintaining organization of an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.8	The system shall allow a user with the View TSS Configuration right to	TSS	View TSS Details	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	view the operations center configured to receive hardware failure alerts for an internal TSS.			
SR1.5.7.5.2.9	The system shall allow a user with the View TSS Configuration right to view the polling interval for an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.10	The system shall allow a user with the View TSS Configuration right to view whether device logging is enabled for an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.11	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the location of the TSS, in accordance with SR1.5.7.8 View Device Location.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12	The system shall allow a user with the View TSS Configuration right to view the communications settings for an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12.1	The system shall allow a user with the View TSS Configuration right to view the communications port type used to communicate with an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12.2	The system shall allow a user with the View TSS Configuration right to view the port manager timeout for an internal TSS when the TSS is configured to use a modem port or direct port for communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12	The system shall allow a user with	TSS	View TSS Details	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
.3	the View TSS Configuration right to view serial port configuration parameters (baud rate, data bits, stop bits, parity, and flow control for an internal TSS when the TSS is configured to use a modem port or direct port for communications.			
SR1.5.7.5.2.12 .4	The system shall allow a user with the View TSS Configuration right to view the modem related communications settings of an internal TSS when the TSS is configured to use a modem port for communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12 .4.1	The system shall allow a user with the View TSS Configuration and View TSS Sensitive Config rights to view the default phone number for an internal TSS when the TSS is configured to use a modem port for communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12 .4.2	The system shall allow a user with the View TSS Configuration and View TSS Sensitive Config rights to view the port managers and associated phone numbers for an internal TSS when the TSS is configured to use a modem port for communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12 .5	The system shall allow a user with the View TSS Configuration and View TSS Sensitive Config rights to view the port managers and	TSS	View TSS Details	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	associated port names for an internal TSS when the TSS is configured to use a direct port for communications.			
SR1.5.7.5.2.12.6	The system shall allow a user with the View TSS Configuration right to view the TCP/IP related communications settings of an internal TSS when the TSS is configured to use TCP/IP communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.12.6.1	The system shall allow a user with the View TSS Configuration and View TSS Sensitive Config rights to view the IP Address and Port for an internal TSS when the TSS is configured to use TCP/IP communications.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.13	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the zone group configuration of the TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.13.1	The system shall allow a user with the View TSS Configuration right (plus View External TSS right, if the TSS is external) to view the description, direction of travel, posted speed, bearing (towards or away from TSS bearing), and detection zones comprising each zone group.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.2.14	The system shall allow a user with	TSS	View TSS Details	



Tag	Requirement	Feature	Use Cases	Other Design Elements
	the View TSS Configuration right to view whether an internal TSS is configured to run Built-in Test (BIT) at the scheduled time of day or not. (The scheduled time of day is specified in the System Profile. Does not apply to external TSSs.)			
SR1.5.7.5.3	The system shall allow a suitably privileged user to view the status information of a TSS in the system.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.1	The system shall allow any user to view status information of an internal TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.2	The system shall allow a user with the View External TSS right to view status information of an external TSS.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.3	The system shall allow any user to view the communications mode (online, offline, maint mode) of an internal TSS, and of an external TSS if the user has the View External TSS right.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.4	The system shall allow any user to view the operational status (OK, comm failed, hardware failed) of an internal TSS, and of an external TSS if the user has the View External TSS right.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.5	The system shall allow any user to view the hardware failure details of an internal TSS, and of an external TSS if the user has the View External	TSS	View TSS Details	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	TSS right.			
SR1.5.7.5.3.5.1	Hardware failure details for an RTMS model of internal TSS shall include the time Built-in Test (BIT) was last executed on the RTMS (if ever).	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.5.2	Hardware failure details for an RTMS model of internal TSS shall include the time Built-in Test (BIT) most recently failed on the RTMS (if ever).	TSS		
SR1.5.7.5.3.5.3	Hardware failure details for an RTMS model of internal TSS shall include a list of the faults encountered by the last executed BIT (if any)	TSS		
SR1.5.7.5.3.6	The system shall allow any user to view the last status time of a TSS, and of an external TSS if the user has the View External TSS right.	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.5.3.7	The system shall allow a user with the View VSO Summary Data right for a TSS (plus View External TSS right, if the TSS is external) to view a speed range for zone groups of that TSS.	TSS	View TSS Traffic Data	(existing fcn now incl G4)
SR1.5.7.5.3.8	The system shall allow a user with the View VSO Detail Data right for a TSS (plus View External TSS right, if the TSS is external) to view exact speed data for zone groups of that TSS.	TSS	View TSS Traffic Data	(existing fcn now incl G4)
SR1.5.7.5.3.9	The system shall allow a user with the View VSO Detail Data right for a TSS (plus View External TSS right, if the TSS is external) to view volume	TSS	View TSS Traffic Data	(existing fcn now incl G4)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	data for zone groups of that TSS.			
SR1.5.7.5.3.10	The system shall allow a user with the View VSO Detail Data right for a TSS (plus View External TSS right, if the TSS is external) to view occupancy data for zone groups of that TSS.	TSS	View TSS Traffic Data	(existing fcn now incl G4)
SR1.5.7.8	View Device Location	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.8.1	The location information displayed for a device shall include a location description, latitude/longitude, state, county or region, route type, route number (or name, and flag to indicate which to use), direction (including bidirectional directions), proximity to intersecting feature (if any), and intersecting feature information (if any).	TSS	View TSS Details	(existing fcn now incl G4)
SR1.5.7.8.1.1	When displaying location information for a device, intersecting feature information shall include intersecting route type, route number (or name, and flag to indicate which to use), state or county milepost, or available exit information.	TSS	View TSS Details	(existing fcn now incl G4)
SR3.1.2.1.7	The system shall poll each online internal CHART detector at its configured polling interval.	TSS	Poll TSS	
SR3.1.2.1.7.1	CHART shall poll CHART detectors configured to be reached at the same endpoint together, serially, as necessary to ensure each can be	TSS	Poll TSS	RTMSFactoryImp:pollDevices.etd RTMSImpl:poll.etd TSSPollingTask:run.etd

Tag	Requirement	Feature	Use Cases	Other Design Elements
	successfully polled without interference from the other(s).			
SR3.1.2.1.7.1.1	CHART shall schedule polls of TSS detectors reachable via the same port manager at the same phone number together, and poll each serially (one after another) on the same phone call, to avoid interference between these detectors. (Such detectors would necessarily have different drop addresses.)	TSS	Poll TSS	RTMSFactoryImp:pollDevices.etd RTMSImpl:poll.etd TSSPollingTask:run.etd
SR3.1.2.1.7.1.2	CHART shall schedule polls of TSS detectors reachable via the same IP address and the same TCP port together, and poll each serially (one after another) on the same TCP connection, to avoid interference between these detectors. (Such detectors would necessarily have different drop addresses.)	TSS	Poll TSS	RTMSFactoryImp:pollDevices.etd RTMSImpl:poll.etd TSSPollingTask:run.etd
SR3.1.2.1.7.2	On each poll of each detector, the system shall query the detector for Average Speed data accumulated since the last poll, for each zone (lane).	TSS	Poll TSS	
SR3.1.2.1.7.3	On each poll of each detector, the system shall query the detector for Occupancy Percentage data accumulated since the last poll, for each zone (lane).	TSS	Poll TSS	
SR3.1.2.1.7.4	On each poll of each detector, the system shall query the detector for	TSS	Poll TSS	

Tag	Requirement	Feature	Use Cases	Other Design Elements
	Volume Count data accumulated since the last poll, for each zone (lane).			
SR3.2.2	The system shall automatically send the recorded data from internal CHART detectors and sensors to the archive after the pre-determined time.	TSS	Archive TSS Data	
SR3.2.2.1	The system shall archive all detector traffic monitoring data per detection zone.	TSS	Archive TSS Data	
SR3.2.2.1.1	If the software hosting TSS devices is restarted, each internal TSS object shall re-load its most recently acquired traffic data from the data store, provided that data is within a configurable age. (The age is the minimum of a DBA-configurable archive job parameter and a software maintainer configurable startup property.)	TSS	Load TSS Data	
SR3.6.1.1.3.2	The system shall prevent flash video streaming if a camera image has been blocked from public viewing, for any Streaming Flash Servers configured for the camera that are designated as "public".	Video	Block Display To Public	GUI: VideoSessionReqHdlr : updateVideoSessionXML SD, Server: CameraControlModule : BlockToPublic SD, VideoCameraImpl : setStreamsBlockedInPublicSFSs SD
SR3.6.1.1.4.1	The system shall automatically enable flash video streaming as a camera image is unblocked from public viewing for all Streaming Flash Servers configured for the camera that are designated as	Video	Unblock Camera Display To Public	GUI: none (unchanged for R9), Server: VideoCameraImpl: unblockToPublic SD, VideoCameraImpl: setStreamsBlockedInPublicSFSs SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	"public".			
SR3.6.1.4.1	The system shall allow a camera to be controlled only if the camera image is displayed on a local monitor or in a video session on the user's desktop.	Video	Request Camera Control	VideoAndControlData : requestControlSession SD, VideoAndControlData : videoRequiredForControl SD, VideoAndControlData : requestVideoSession SD
SR3.6.1.4.1.2	The system shall automatically open a desktop video session when a user requests (or overrides) control, if the camera is not already displayed on a local monitor or on the user's desktop, if the user has the View Desktop Video right and the camera is eligible for desktop video display.	Video	Request Camera Control, Override Camera Control	VideoAndControlData : requestControlSession SD, VideoAndControlData : videoRequiredForControl SD, VideoAndControlData : requestVideoSession SD
SR3.6.1.4.1.2.1	The system shall automatically play the video, if it automatically acquired a video session to ensure that the user has the camera displayed locally.	Video	Request Camera Control, Override Camera Control	none
SR3.6.1.4.1.3	The system shall allow an operator controlling a camera to close a desktop video session (without closing the control session) only if the camera is displayed on a local monitor.	Video	View Video And Camera Control Dialog	VideoAndControlData : releaseVideoSession SD
SR3.6.1.4.17.2	The system shall allow a camera's control to be overridden only if the camera image is displayed on a local monitor or in a video session on the user's desktop.	Video	Override Camera Control	VideoAndControlData : requestControlSession SD, VideoAndControlData : videoRequiredForControl SD, VideoAndControlData : requestVideoSession SD
SR3.6.3.9.1	The system shall allow a suitably privileged operator to revoke a camera image from monitors and	Video	Block Display To Public	GUI: none (block to public - unchanged for R9); ControlVideoSourceReqHdlr : processSetSFSBlocked SD (block to SFS);

Tag	Requirement	Feature	Use Cases	Other Design Elements
	desktop video sessions.			VideoSessionReqHdlr : updateVideoSessionXML SD (block to org), Server: CameraControlModule : BlockToPublic SD, VideoCameraImpl : setStreamsBlockedInPublicSFSs SD, VideoCameraImpl : setSFSBlocked SD
SR3.6.3.13.2.1	The system shall automatically stop flash video streaming when a camera image is removed from public viewing for any Streaming Flash Servers configured for the camera that are designated as "public".	Video	Block Camera Display To Public	GUI: none (unchanged for R9) Server: CameraControlModule : BlockToPublic SD, VideoCameraImpl : setStreamsBlockedInPublicSFSs SD
SR3.6.3.13.3	The system shall allow a user with the Block Display To Web And Media right to block (stop and prevent) the display of a camera to a specified Streaming Flash Server that is configured for the camera.	Video	Block Camera Display To Flash Server	GUI: ControlVideoSourceReqHdlr : processSetSFSBlocked SD, Server: VideoCameraImpl : setSFSBlocked SD
SR3.6.3.13.3.1	The system shall allow a user with the Block Display To Web And Media right to unblock the display of a camera to a specified Streaming Flash Server that is configured for the camera.	Video	Unblock Camera Display From Flash Server	GUI: ControlVideoSourceReqHdlr : processSetSFSBlocked SD, Server: VideoCameraImpl : setSFSBlocked SD
SR3.6.6.4.9	The system shall display all desktop video sessions displaying the tour list.	Video	View Tour Details	VideoTourReqHdlr : viewTourListDetails SD, MiscDataClasses CD
SR3.6.6.4.9.1	The system will display the name of the user who is displaying the video session.	Video	View Tour Details	VideoTourReqHdlr : viewTourListDetails SD, MiscDataClasses CD
SR3.6.6.4.9.2	The system will display the operations center of the user who is	Video	View Tour Details	VideoTourReqHdlr : viewTourListDetails SD, MiscDataClasses CD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	displaying the video session.			
SR3.6.6.4.9.3	The system will display the host name and/or IP address of the computer that is displaying the video session.	Video	View Tour Details	VideoTourReqHdlr : viewTourListDetails SD, MiscDataClasses CD
SR3.6.6.14.7	The system shall indicate to the user if a tour list is being displayed in a video session on the user's desktop.	Video	View Video Source List	ViewVideoSourceReqHdlr : processViewVideoSourceListReq SD, ServletBaseClasses CD, MiscDataClasses CD
SR3.6.9.2.1	The system shall allow a user to display a streaming camera image on his or her desktop monitor, if a Streaming Flash Server is configured for the camera that matches the GUI's designated Streaming Flash Server, subject to restrictions listed in sub-requirements.	Video	View Desktop Video For Source, View Video And Camera Control Dialog	VideoSessionReqHdlr : viewVideoSession SD, VideoSessionReqHdlr : updateVideoSessionXML SD, VideoDisplayComponent : handleUpdateSessionResponse SD
SR3.6.9.2.1.1	The system shall allow desktop video to be displayed only if the user has the View Desktop Video right.	Video	View Desktop Video	VideoSessionReqHdlr : viewVideoSession SD
SR3.6.9.2.1.2	The system shall allow desktop video to be displayed only if the camera is online.	Video	View Desktop Video	VideoSessionReqHdlr : updateVideoSessionXML SD
SR3.6.9.2.1.3	The system shall allow desktop video to be displayed only if the camera is not blocked for the organization assigned to the operations center at which the user is logged in.	Video	View Desktop Video	VideoSessionReqHdlr : updateVideoSessionXML SD
SR3.6.9.2.1.4	The system shall allow desktop video to be displayed only if the camera is not blocked to the GUI's designated Streaming Flash Server.	Video	View Desktop Video	VideoSessionReqHdlr : updateVideoSessionXML SD
SR3.6.9.2.1.6	The system shall allow desktop	Video	Acquire Video	GUI: VideoSessionReqHdlr : viewVideoSession



Tag	Requirement	Feature	Use Cases	Other Design Elements
	video to be displayed for a video source only if the number of existing desktop video sessions for the user's operations center is less than the center's limit.		Session	SD, Server: OperationsCenterImpl : requestVideoSession SD
SR3.6.9.2.1.7	The system shall indicate when a video source is no longer eligible for display on the desktop (e.g., blocked for the user's op center's organization, blocked to the GUI's designated Streaming Flash Server, or if the camera goes offline).	Video	Monitor Video Source And Session Status	VideoSessionReqHdlr : updateVideoSessionXML SD, VideoDisplayComponent : handleUpdateSessionResponse SD
SR3.6.9.2.2	The system shall display the name of the video source for the desktop video.	Video	View Desktop Video For Source, View Video And Camera Control Dialog	none (prototype only)
SR3.6.9.2.3	The system shall display a description of the location of the video source for the desktop video.	Video	View Desktop Video For Source, View Video And Camera Control Dialog	none (prototype only)
SR3.6.9.3.1	The system shall allow a user with the View Desktop Video right to display a tour of streaming camera images on his or her desktop, if the number of active video sessions is less than the operations center's assigned limit.	Video	View Desktop Video Tour	VideoSessionReqHdlr : viewVideoSession SD
SR3.6.9.3.2	The system shall allow the user to select a preconfigured video tour for display on the desktop.	Video	View Desktop Video Tour	VideoSessionReqHdlr : viewVideoSession SD
SR3.6.9.3.2.2	The system shall ignore any camera presets defined in the tour when displaying on the desktop. (NOTE -	Video	Play Desktop Video	N/A

Tag	Requirement	Feature	Use Cases	Other Design Elements
	this is to avoid issues that could arise from too many desktop tours trying to command the camera).			
SR3.6.9.3.2.2.1	The system shall indicate to the user that presets will be ignored when playing the preconfigured video tour on the desktop.	Video	View Desktop Video Tour, View Tour Details	none
SR3.6.9.3.2.4	The system shall display the name of the preconfigured tour when displaying the tour in a desktop video session.	Video	View Desktop Video Tour	none (prototype only)
SR3.6.9.3.3	The system shall stop displaying a camera in a desktop video tour if the camera becomes ineligible for display after the tour is already running.	Video	Monitor Video Source And Session Status	VideoDisplayComponent : handleUpdateSessionResponse SD, VideoDisplayComponent : displayNextEligibleVideoSourceInTour SD
SR3.6.9.3.4	The system shall include a camera in a desktop video tour if the camera becomes eligible for display after the tour is already running.	Video	Monitor Video Source And Session Status	VideoDisplayComponent : handleUpdateSessionResponse SD, VideoDisplayComponent : displayNextEligibleVideoSourceInTour SD, VideoDisplayComponent : handleTourTimer SD
SR3.6.9.3.7	The system shall indicate if all video sources in the tour become ineligible to be displayed on the desktop.	Video	Monitor Video Source And Session Status	VideoSessionReqHdlr : updateVideoSessionXML SD, VideoDisplayComponent : handleUpdateSessionResponse SD
SR3.6.9.4.2.1	The system shall allow a user to play video within a video session if the video was previously paused or not already playing.	Video	Play Desktop Video	VideoDisplayComponent : playAndPause SD
SR3.6.9.4.2.1.1	The system shall resume video playback as close to the current time as possible, if the video was previously paused.	Video	Play Desktop Video	VideoDisplayComponent : playAndPause SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR3.6.9.4.2.2	The system shall allow a user to pause video within a video session, if not already paused.	Video	Pause Desktop Video	VideoDisplayComponent : playAndPause SD
SR3.6.9.4.2.2.1	The system shall pause the switching of video sources when a desktop tour is paused, so that the image that was displayed at the time of pausing remains displayed.	Video	Pause Desktop Video	VideoDisplayComponent : handleTourTimer SD
SR3.6.9.4.2.3	The system shall allow the user to resize the video image that is displayed in a video session.	Video	Resize Desktop Video	none (prototype only)
SR3.6.9.4.2.3.1	The system shall allow the user to display the video in full screen mode within a video session.	Video	Resize Desktop Video	none (prototype only)
SR3.6.9.4.2.3.1.1	The system shall use only one monitor head when displaying full screen video on a computer with more than one monitor.	Video	Resize Desktop Video	none (prototype only)
SR3.6.9.4.2.3.1.2	The system shall allow the user to exit full screen mode when displaying full screen video.	Video	Resize Desktop Video	none (prototype only)
SR3.6.9.4.2.3.1.2.1	The system shall cause the video to be displayed at the size that was used prior to invoking full screen mode, when exiting full screen mode.	Video	Resize Desktop Video	none (prototype only)
SR3.6.9.4.2.4	The system shall allocate a desktop video session resource from the user's operations center's limit before showing a video stream on the desktop, reducing the number of available video sessions for that operations center by one.	Video	Acquire Video Session	GUI: VideoSessionReqHdlr : viewVideoSession SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR3.6.9.4.2.4.1	The system shall display an appropriate error message to the user if a requested video session could not be obtained due to exceeding the operations center's limit.	Video	Acquire Video Session	VideoSessionReqHdlr : viewVideoSession SD
SR3.6.9.4.2.5	The system shall end (or release) the video session resource after the user closes the window that is displaying the corresponding video (or closes the desktop video portion of the camera control window) .	Video	End Video Session	VideoSessionReqHdlr : endOwnVideoSession SD
SR3.6.9.4.2.7	The system shall stop playing video in a video session if the session was canceled by another user.	Video	Monitor Video Source And Session Status	VideoSessionReqHdlr : endUserVideoSession SD, ResourceMgmtPushConsumer : handleVideoSessionEnded SD, VideoSessionReqHdlr : updateVideoSessionXML
SR3.6.9.4.2.7.1	The system shall display a message to the user explaining why the video was stopped, if the user's video session was canceled by another user.	Video	Monitor Video Source And Session Status	VideoSessionReqHdlr : endUserVideoSession SD, ResourceMgmtPushConsumer : handleVideoSessionEnded SD, VideoSessionReqHdlr : updateVideoSessionXML
SR3.6.9.4.3	The system shall allow the user to have multiple desktop video sessions playing simultaneously on the same computer.	Video	View Desktop Video	none
SR3.6.9.4.5	The system shall allow a user with the Cancel Video Session right to cancel another user's video session.	Video	Cancel Video Session For User	GUI: VideoSessionReqHdlr : endUserVideoSession SD, ResourceMgmtPushConsumer : handleVideoSessionEnded SD, Server: OperationsCenterImpl : endVideoSession SD
SR3.6.9.5.1	The system shall allow the user to view the list of desktop video sessions that exist in the system.	Video	View Video Sessions	GUIVideoServletClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR3.6.9.5.1.1	The detailed data displayed for a desktop video session shall include the login name of the user.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.2	The detailed data displayed for a desktop video session shall include the name of the camera or tour displaying on the desktop.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.3	The detailed data displayed for a desktop video session shall include the hostname and/or IP of the desktop displaying the video session.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.4	The detailed data displayed for a desktop video session shall include the hostname and/or IP of the session provider.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.5	The detailed data displayed for a desktop video session shall include the operation center the user is logged in to.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.6	The detailed data displayed for a desktop video session shall include the date/time when the user started the session.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.1.8	The detailed data displayed for a desktop video session shall include the date/time when the session was last confirmed to be in use.	Video	View Video Sessions	MiscDataClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2	The system shall allow the user to sort the desktop video sessions list	Video	View Video Sessions	GUIVideoServletClasses CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.1	The system shall allow the user to sort the desktop video sessions list by login name.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR3.6.9.5.2.2	The system shall allow the user to sort the desktop video sessions list by camera or tour name.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.3	The system shall allow the user to sort the desktop video sessions list by hostname or IP address of the desktop.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.4	The system shall allow the user to sort the desktop video sessions list by hostname or IP address of the session provider.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.5	The system shall allow the user to sort the desktop video sessions list by the operation center where the user logged in.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.6	The system shall allow the user to sort the desktop video sessions list by the timestamp when the user started the session.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.2.7	The system shall allow the user to sort the desktop video sessions by the date/time when the session was last confirmed to be in use.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3	The system shall allow the user to filter the desktop video sessions list	Video	View Video Sessions	GUIVideoServletClasses2 CD, VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.1	The system shall allow the user to filter the desktop video sessions list by login name.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.2	The system shall allow the user to filter the desktop video sessions list by camera or tour name.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.3	The system shall allow the user to filter the desktop video sessions list	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	by hostname or IP address of the desktop.			
SR3.6.9.5.3.4	The system shall allow the user to filter the desktop video sessions list by hostname or IP address of the session provider.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.5	The system shall allow the user to filter the desktop video sessions list by the operation center where the user logged in.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.6	The system shall allow the user to filter the desktop video sessions list by the timestamp when the user started the session.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.3.7	The system shall allow the user to filter the desktop video sessions by the date/time when the session was last confirmed to be in use.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR3.6.9.5.4	The desktop video session list shall initially contain the current user's desktop video sessions.	Video	View Video Sessions	VideoSessionReqHdlr: viewVideoSessionList SD
SR4.2.3.2.1.2	The system shall consider all applicable devices located within a configurable radius of the traffic event.	Decision Support	MapAndGISUses.ViewCloseDevicesOnMap MapAndGISUses.ViewDevicesCloseToTrafficEvent	
SR4.2.3.2.1.2.3.1	The system shall suggest messages for CHART DMS devices within a configurable radius of a CHART user created traffic event.	Decision Support	Respond to Traffic Event Use Case: Request Suggested Response Actions	Screenshot: HMI Figure 2 DecisionSupport CD TrafficEventManager CD TrafficEventManager2 CD TrafficEventModuleClasses2 CD TrafficEventModule2:initialize SD DecisionSupportManager:getDMSList SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR4.2.3.2.1.2.3.1.1	The system shall determine the radius using the configured distance categories and the percentage of lanes closed for the Traffic Event.	Decision Support		
SR4.2.3.2.1.2.3.1.1.1	The system shall determine lane closed percentage for directional traffic events (I.E. northbound, southbound) using the percentage of travel lanes in the direction of the traffic event that are closed.	Decision Support		
SR4.2.3.2.1.2.3.1.1.2	The system shall determine lane closed percentage for non-directional traffic events (I.E. other, north/south) using the percentage of all travel lanes that are closed.	Decision Support		
SR4.2.3.2.1.2.3.1.2	The system shall determine the radius, for traffic event types not supporting lane status, using the max distance of the FAR distance category (Note: See Req 4.2.3.2.9.3 for distance category configuration).	Decision Support		
SR4.2.3.2.1.2.9	The system shall not recommend devices whose calculated proximity to the target traffic event is a proximity that is configured to have suggestions disabled (Note: See Req 4.2.3.2.9.8 for proximity related configuration).	Decision Support		
SR4.2.3.2.1.2.10	The system shall recommend 0 or more messages for each DMS that meets all device usage criteria.	Decision Support	Respond to Traffic Event Use Case: Request Suggested DMS Messages	Screenshot: HMI Figure 3 Screenshot: HMI Figure 4 DecisionSupportUtility CD DecisionSupportManager:applyTemplate SD DecisionSupportManager:generateSuggestions ForDMS SD DecisionSupportDataClasses CD



Tag	Requirement	Feature	Use Cases	Other Design Elements
				chartlite.data.trafficevents.DecisionSupport CD chartlite.data.trafficevents_classes CD chartlite.servlet.trafficevents_classes CD TrafficEventGroup:requestSuggestions SD ResponsePlanReqHdlr:getSuggDMSActions SD ResponsePlanReqHdlr:viewSuggActionsCommandStatus SD ResponsePlanReqHdlr:viewSuggDMSActions SD
SR4.2.3.2.1.2.10.1	The system shall create DMS message suggestions based on configured message templates.	Decision Support	Respond to Traffic Event Use Case: Request Suggested DMS Messages	DecisionSupportManager:applyTemplate SD DecisionSupportManager:generateSuggestionsForDMS SD ResponsePlanReqHdlr:getSuggDMSActions SD ResponsePlanReqHdlr:viewSuggActionsCommandStatus SD ResponsePlanReqHdlr:viewSuggDMSActions SD
SR4.2.3.2.1.2.10.1.1	The system shall create a DMS message suggestion for each message template that matches the DMS proximity type, distance category, geometry and the type of the traffic event provided that the template tags can successfully be substituted with data from the traffic event as specified in the immediate sub-requirements.	Decision Support	Respond to Traffic Event Use Case: Request Suggested DMS Messages	ResponsePlanReqHdlr:getSuggDMSActions SD ResponsePlanReqHdlr:viewSuggActionsCommandStatus SD ResponsePlanReqHdlr:viewSuggDMSActions SD
SR4.2.3.2.1.2.10.1.1.1	The system shall disregard a message template when generating message suggestions for a traffic event if the message template contains a route name parameter tag and the system has been configured to not show route names for the route type of the route the	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	traffic event is located on.			
SR4.2.3.2.1.2.10.1.1.2	The system shall disregard a message template when generating message suggestions for an Event if the message template contains a route type and/or route number parameter tag and the system has been configured to not show route type/number for the Event's route type.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.3	The system shall disregard a message template when generating message suggestions for an Event if the message template contains a route direction parameter tag but no replacement value for the Event's route direction type has been configured in the system.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.4	The system shall generate a DMS message suggestion by replacing all words/phrases configured in the word substitutions list and all parameter tags specified in a message template with appropriate DMS and Event values.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.4.1	The system shall use long replacement values for word/phrase substitutions and parameter tags that support long values when generating the message suggestion.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.4.1.1	The system shall regenerate the message suggestion using short	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	replacement values for word/phrase substitutions and parameter tags that support short values if the generated message does not fit on the target DMS when using long values.			DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.4.2	The system shall remove a word/phrase from the message suggestion, leaving no extraneous blanks, if both the short and long values configured for the word/phrase are blank.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2.10.1.1.4.3	The system shall generate a lane closure description which will be used as the replacement value for templates containing the current lane closures parameter tag.	Decision Support		DSTemplateTag.getMulti SD Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2.10.1.1.4.3.1	The system shall generate a lane closure description that describes the lanes affected in the direction of the traffic event.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2.10.1.1.4.3.2	The system shall include only travel lanes when generating a lane closure description for a directional event (N, S, E, W, IL, OL) if 1 or more travel lanes are currently closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2.10.1.1.4.3.2.1	The lane closure description will be "LEFT LANE " and configured closure word for the event type (see SR4.2.3.2.9.10), if the leftmost travel lane is closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2.10.1.1.4.3.2.2	The lane closure description will be "RIGHT LANE " and configured	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	closure word for the event type (see SR4.2.3.2.9.10), if the rightmost travel lane is closed.			
SR4.2.3.2.1.2. 10.1.1.4.3.2.3	The lane closure description will be "ALL LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if all travel lanes are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.2.4	The lane closure description will be "n LEFT LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if multiple contiguous travel lanes are closed and are grouped all the way to the left (n = number of lanes closed).	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.2.5	The lane closure description will be "n RIGHT LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if multiple contiguous travel lanes are closed and are grouped all the way to the right (n = number of lanes closed).	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.2.6	The lane closure description will be "n OF x LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if travel lane closures are non-contiguous or are contiguous but not grouped all to the left or all to the right (n = closed lanes, x = total lanes).	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.3	The system shall only include exit lanes when generating a lane closure description for a directional	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	Event (N, S, E, W, OL, IL) if no travel lanes are currently closed and at least one exit lane is closed.			
SR4.2.3.2.1.2. 10.1.1.4.3.3.1	The lane closure description will be "LEFT EXIT " and configured closure word for the event type (see SR4.2.3.2.9.10), if all left exit lanes are closed and no right exit lanes are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.3.2	The lane closure description will be "RIGHT EXIT " and configured closure word for the event type (see SR4.2.3.2.9.10), if all right exit lanes are closed and no left exit lanes are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.3.3	The lane closure description will be "n OF x LEFT EXIT LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if some left exit lanes are closed and no right exit lanes are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.3.4	The lane closure description will be "n OF x RIGHT EXIT LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if some right exit lanes are closed and no left exit lanes are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.3.5	The lane closure description will be "n OF x EXIT LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if both left and right exit lane closures exist but not all are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR4.2.3.2.1.2. 10.1.1.4.3.3.6	The lane closure description will be "ALL EXITS " and configured closure word for the event type (see SR4.2.3.2.9.10), if both left and right exit lanes exist and all are closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.4	The system shall only include shoulders when generating a lane closure description for a directional Event (N, S, E, W, OL, IL) if no travel lanes or exit lanes are currently closed and at least one shoulder is closed.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.4.1	The lane closure description will be "LEFT SHOULDER " and configured closure word for the event type (see SR4.2.3.2.9.10), if shoulder lane closures are contiguous and grouped to the left.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.4.2	The lane closure description will be "RIGHT SHOULDER " and configured closure word for the event type (see SR4.2.3.2.9.10), if shoulder lane closures are contiguous and grouped to the right.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.4.3	The lane closure description will be "SHOULDERS " and configured closure word for the event type (see SR4.2.3.2.9.10), if shoulder lane closures are non-contiguous.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.5	The system shall include only travel lanes when generating a lane closure description for bi-directional (N/S, E/W, OL/IL) events.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR4.2.3.2.1.2. 10.1.1.4.3.5.1	The lane closure description will be "n OF x LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if some but not all travel lanes are closed (considering travel lanes on both directions of the lane configuration).	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.5.2	The lane closure description will be "ALL LANES " and configured closure word for the event type (see SR4.2.3.2.9.10), if all travel lanes are closed in both directions.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.4.3.6	The system shall generate a lane closure description of "ALL LANES OPEN" if the Event does not include a lane status or the lane status indicates no closures of travel lanes, exit lanes or shoulders.	Decision Support		Lane Closure Description Generation Flow Chart (Appendix)
SR4.2.3.2.1.2. 10.1.1.5	The system shall disregard a suggested message (after all tag replacement and all word substitutions / abbreviations have been done) if the message does not fit on the target DMS.	Decision Support		DMSDecSupMsgTemplateModel.formatMulti SD DMSDecSupMsgTemplateRow.formatMultiPublic SD DMSDecSupMsgTemplateRow.formatMultiPrivate SD DSTemplateTag.getMulti SD
SR4.2.3.2.1.2. 10.2	The system shall recommend DMS Messages by finding plan items that target the DMS and are contained within plans that are configured to pertain to the specified traffic event type.	Decision Support	Respond to Traffic Event Use Case: Request Suggested Plans	Screenshot: HMI Figure 13 DecisionSupportDataClasses CD chartlite.data.trafficevents.DecisionSupport CD chartlite.data.trafficevents_classes CD TrafficEventModule2:initialize SD TrafficEventGroup:requestSuggestions SD
SR4.2.3.2.1.2. 10.3	The system shall present all suggested messages for a DMS ordered by a score (highest to	Decision Support	Respond to Traffic Event Use Case: Request Suggested	Screenshot: HMI Figure 3 Screenshot: HMI Figure 8 DecisionSupportDataClasses CD chartlite.data.trafficevents.DecisionSupport CD

Tag	Requirement	Feature	Use Cases	Other Design Elements
	lowest) based on specificity of the message suggestion.		DMS Messages	chartlite.data.trafficevents_classes CD DecisionSupportManager.applyTemplates SD
SR4.2.3.2.1.2. 10.3.1	The system shall score a suggestion generated from a template with fewer supported event types higher than one with more supported event types (all other criteria being the same).	Decision Support		
SR4.2.3.2.1.2. 10.3.2	The system shall score a suggestion generated from a template with more stringent proximity requirements (same route, same dir, upstream) higher than one with less stringent proximity requirements (all other criteria being the same).	Decision Support		
SR4.2.3.2.1.2. 10.3.3	The system shall score a suggestion generated from a template with fewer supported distance categories higher than one with more supported distance categories (all other criteria being the same).	Decision Support		
SR4.2.3.2.1.2. 10.3.4	The system shall score a suggestion generated from a template with fewer supported sign geometries higher than one with more supported sign geometries (all other criteria being the same).	Decision Support		
SR4.2.3.2.1.2. 10.3.5	The system shall score a suggestion generated from a template with fewer pages higher than one with more pages (all other criteria being the same).	Decision Support		
SR4.2.3.2.1.2.	The system shall score a suggestion	Decision		



Tag	Requirement	Feature	Use Cases	Other Design Elements
10.3.6	generated from a template with more parameter tags higher than one with less parameter tags (all other criteria being the same).	Support		
SR4.2.3.2.1.2. 10.3.7	The system shall display summary scoring information about a message suggestion indicating the number of parameter tags and template filters (ex. proximity, distance category) matched during scoring.	Decision Support		Screenshot: HMI Figure 8
SR4.2.3.2.1.2. 10.3.7.1	The system shall display detailed scoring information about a message suggestion, if requested by the user, indicating which parameter tags and template filters matched during scoring.	Decision Support		Screenshot: HMI Figure 8
SR4.2.3.2.1.2. 10.4	The system shall display the current active DMS message for any DMS with message suggestions, if requested by the user.	Decision Support		Screenshot: HMI Figure 7
SR4.2.3.2.1.5	The system shall allow an operator to request suggestions for devices currently in a traffic event response plan.	Decision Support	Use Case: Request Suggested Message for Response Plan Devices	Screenshot: HMI Figure 17 chartlite.data.trafficevents_classes CD
SR4.2.3.2.1.5. 1	The system shall allow the operator to request suggested messages for all DMS devices currently in a traffic event response plan.	Decision Support	Respond to Traffic Event Use Case: Request Suggested Message for Response Plan Devices	Screenshot: HMI Figure 17
SR4.2.3.2.1.5. 2	The system shall allow the operator to request suggested messages for	Decision Support	Respond to Traffic Event Use Case:	Screenshot: HMI Figure 17

Tag	Requirement	Feature	Use Cases	Other Design Elements
	only the DMS devices currently in a traffic event response plan that have no planned message.		Request Suggested Message for Response Plan Devices	
SR4.2.3.2.1.8	The system shall allow a user with proper rights to disable the suggestion of a particular device or device plan.	Decision Support	Respond to Traffic Event Use Case: Disable Suggestions For A DMS Respond to Traffic Event Use Case: Disable Suggestions for a Plan	Screenshot: HMI Figure 9 Screenshot: HMI Figure 12 TrafficEventGroup:disableSuggestionsFor SD
SR4.2.3.2.1.8.1	The user shall be allowed to indicate that suggestions for the device or plan should be disabled for the life of the traffic event, or only for the current session.	Decision Support		
SR4.2.3.2.1.8.2	The user shall be allowed to re-enable suggestions for a device or plan that has previously been disabled for suggestions.	Decision Support		Screenshot: HMI Figure 9 Screenshot: HMI Figure 12 TrafficEventGroup:enableSuggestionsFor SD
SR4.2.3.2.9	The system shall allow a user with the configure system right to configure decision support settings	Decision Support		Screenshot: HMI Figure 23 chartlite.servlet.usermgmt.systemProfile_classes CD DecisionSupportSystemProfileReqHdlr:getDecisionSupportSettingsForm SD DecisionSupportSystemProfileReqHdlr:setDecisionSupportSettings SD
SR4.2.3.2.9.2	A user with the configure system right shall be allowed to maintain a system wide list of approved word substitutions and abbreviations to be utilized when suggesting device	Decision Support	Configure Decision Support Use Case: Configure Word Substitution	Screenshot: HMI Figure 31 Screenshot: HMI Figure 32

Tag	Requirement	Feature	Use Cases	Other Design Elements
	message content			
SR4.2.3.2.9.2.1	A user shall be able to specify a short value and a long value of the substitution / abbreviation for a word/phase in the list.	Decision Support	Configure Decision Support Use Case: Configure Word Substitution	Screenshot: HMI Figure 31 Screenshot: HMI Figure 32
SR4.2.3.2.9.2.2	A user shall be able to specify blanks for the substitution / abbreviation values in order to remove the word/phrase from the message suggestion.	Decision Support	Configure Decision Support Use Case: Configure Word Substitution	
SR4.2.3.2.9.3	A user with the configure system right shall be permitted to configure the decision support message template distance category filters	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureDistances	Screenshot: HMI Figure 24
SR4.2.3.2.9.3.1	A user shall be able to specify the maximum distance from a traffic event for which a template with a distance category of IMMEDIATE shall be used.	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureDistances	Screenshot: HMI Figure 24
SR4.2.3.2.9.3.2	A user shall be able to specify the maximum distance from a traffic event for which a template with a distance category of NEAR shall be used.	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureDistances	Screenshot: HMI Figure 24
SR4.2.3.2.9.3.3	A user shall be able to specify the maximum distance from a traffic	Decision Support	Configure Decision Support Use Case:	Screenshot: HMI Figure 24

Tag	Requirement	Feature	Use Cases	Other Design Elements
	event for which a template with a distance category of FAR shall be used.		ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureDistances	
SR4.2.3.2.9.3.4	A user shall be able to specify a percentage of lanes closed for each distance category for the purpose of device selection (Devices in this distance category should be selected if the target traffic event's lane closure percentage is >= this setting).	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureDistances	Screenshot: HMI Figure 24
SR4.2.3.2.9.4	A user with the Configure System right shall be able to configure route type related settings for decision support.	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureRouteTypes	Screenshot: HMI Figure 26
SR4.2.3.2.9.4.1	A user with the configure system right shall be able to configure a system wide list of route types whose route type and route number should never be used in message content.	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureRouteTypes	Screenshot: HMI Figure 26
SR4.2.3.2.9.4.2	A user with the configure system right shall be able to configure a system wide list of route types whose route name should never be used in message content.	Decision Support	Configure Decision Support Use Case: ConfigureGeneralSettings Configure Decision Support Use Case:	Screenshot: HMI Figure 26

Tag	Requirement	Feature	Use Cases	Other Design Elements
			ConfigureRouteTypes	
SR4.2.3.2.9.5	A user with the configure system right shall be able to configure the word(s) to use in suggested message content for each possible traffic event type.	Decision Support	Configure Decision Support Use Case: ConfigureTagReplaces Configure Decision Support Use Case: ConfigureEventTypeTagReplacements	Screenshot: HMI Figure 33
SR4.2.3.2.9.5.1	A user shall be able to specify both an abbreviated version and a long version of the traffic event type.	Decision Support	Configure Decision Support Use Case: ConfigureTagReplaces Configure Decision Support Use Case: ConfigureEventTypeTagReplacements	Screenshot: HMI Figure 33
SR4.2.3.2.9.6	A user with the configure system right shall be able to configure the word(s) to use in suggested message content for each possible Incident type.	Decision Support	Configure Decision Support Use Case: ConfigureTagReplaces Configure Decision Support Use Case: ConfigureIncidentTypeTagReplacements	Screenshot: HMI Figure 33
SR4.2.3.2.9.6.1	A user shall be able to specify both an abbreviated version and a long version of the incident type.	Decision Support	Configure Decision Support Use Case: ConfigureTagReplaces Configure Decision Support Use Case: ConfigureIncidentTypeTagReplacements	Screenshot: HMI Figure 33

Tag	Requirement	Feature	Use Cases	Other Design Elements
SR4.2.3.2.9.7	A user with the configure system functional right shall be able to maintain a set of message templates that will be used by the system when suggesting message content.	Decision Support	Configure Decision Support Use Case: ConfigureMessageTemplate	GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD
SR4.2.3.2.9.7.1	A user with the configure system right shall be able to maintain a set of message templates to be used by the system when suggesting message content for DMS devices.	Decision Support	Configure Decision Support Use Case: ConfigureDMSMessageTemplate	Screenshot: HMI Figure 27 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD DMSReqHdlr:addDMSDecSuppMsgTemplate SD DMSReqHdlr:editDMSDecSuppMsgTemplate SD MessageTemplateReqHdlr:removeDMSDecSuppMsgTemplate SD GUIDMSServletClasses CD MessageTemplateFactoryImpl.getMsgTemplates SD MessageTemplateFactoryImpl.createMsgTemplate SD DMSTravInfoMsgTemplateImpl.setConfig SD DMSTravInfoMsgTemplateImpl.getConfig SD
SR4.2.3.2.9.7.1.1	A user with the configure system right shall be able to use a DMS message editor to create a DMS message template that consists of static text and parameter tags.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD addDMSDecSuppMsgTemplate SD
SR4.2.3.2.9.7.1.1.1	A user with the configure system functional right shall be able to specify traffic event type as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.2	A user with the configure system functional right shall be able to specify incident type as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.2.1	The system shall automatically set the template event type applicability	Decision Support		

Tag	Requirement	Feature	Use Cases	Other Design Elements
	to incident (and no other types) when the incident type parameter tag is used in the template content.			
SR4.2.3.2.9.7.1.1.3	A user with the configure system functional right shall be able to specify route type as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.4	A user with the configure system functional right shall be able to specify route number as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.5	A user with the configure system functional right shall be able to specify route name as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.6	A user with the configure system functional right shall be able to specify route direction as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.7	A user with the configure system functional right shall be able to specify intersecting exit number (always uses intersecting feature 1) as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.8	A user with the configure system functional right shall be able to specify intersecting exit road name (always uses intersecting feature 1) as a message template parameter tag.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.1.1.9	A user with the configure system functional right shall be able to	Decision Support	Configure Decision Support Use Case:	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29

Tag	Requirement	Feature	Use Cases	Other Design Elements
	specify current lane closures as a message template parameter tag.		AddDMSMessageTemplate	
SR4.2.3.2.9.7.1.1.10	A user with the configure system functional right shall be able to create message templates that contain 2 or less pages.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	
SR4.2.3.2.9.7.1.1.11	A user with configure system right shall be able to create message templates that pertain to DMS geometries that support at least 8 characters per line.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	
SR4.2.3.2.9.7.1.1.12	A user with configure system right shall be able to create message templates that pertain to DMS geometries that support at least 2 lines per page.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate	
SR4.2.3.2.9.7.2	A user with the configure system functional right shall be able to specify the traffic event types that a message template pertains to.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate Configure Decision Support Use Case: EditDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD
SR4.2.3.2.9.7.3	A user with the configure system right shall be able to specify the device proximity types (for example "same route, same dir, upstream" or "other route") that a message template pertains to.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate Configure Decision Support Use Case: EditDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD
SR4.2.3.2.9.7.	A user with the configure system	Decision	Configure Decision	Screenshot: HMI Figure 28 Screenshot: HMI



Tag	Requirement	Feature	Use Cases	Other Design Elements
4	right shall be able to specify the distance categories that a message template pertains to.	Support	Support Use Case: AddDMSMessageTemplate Configure Decision Support Use Case: EditDMSMessageTemplate	Figure 29 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD
SR4.2.3.2.9.7.5	A user with the configure system right shall be able to specify the DMS sign geometries that a message template pertains to.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate Configure Decision Support Use Case: EditDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29 GUIMessageTemplateDataClasses CD GUIMessageTemplateServletClasses CD
SR4.2.3.2.9.7.5.1	The system shall display a preview of the sample message for the template using the smallest selected geometry.	Decision Support	Configure Decision Support Use Case: AddDMSMessageTemplate Configure Decision Support Use Case: EditDMSMessageTemplate	Screenshot: HMI Figure 28 Screenshot: HMI Figure 29
SR4.2.3.2.9.7.5.1.1	The system shall display a warning if the sample message cannot be displayed on any of the selected geometries. Note: Even if the sample message fits it does not guarantee that the same message will fit when substituted with live data values.	Decision Support		Screenshot: HMI Figure 30
SR4.2.3.2.9.8	A user with the configure system right shall be able to specify a list of	Decision Support	Configure Decision Support Use Case:	Screenshot: HMI Figure 25

Tag	Requirement	Feature	Use Cases	Other Design Elements
	device proximity types (for example "same route, same dir, downstream") for which the system should not suggest messages.		ConfigureGeneralSettings Configure Decision Support Use Case: ConfigureProximities	
SR4.2.3.2.9.9	A user with the configure system right shall be able to specify an approved abbreviation for each CHART route direction.	Decision Support	Configure Decision Support Use Case: ConfigureRouteDirectionReplacements	Screenshot: HMI Figure 34
SR4.2.3.2.9.9.1	A user shall be able to specify a blank abbreviation for each CHART route direction (indicating that the route direction should not be used in message content).	Decision Support	Configure Decision Support Use Case: ConfigureRouteDirectionReplacements	
SR4.2.3.2.9.10	A user with the configure system right shall be able to configure the word used to signify closed lanes in the context of a lane closure description for each possible traffic event type.	Decision Support	Configure Decision Support Use Case: ConfigureLaneClosureWord	Not Prototyped
SR4.2.3.2.9.10.1	A user shall be able to specify both an abbreviated version and a long version of the word	Decision Support	Configure Decision Support Use Case: ConfigureLaneClosureWord	Not Prototyped
SR4.2.3.2.9.10.2	The system shall supply a default values of "CLOSED" for Planned Roadway Closures and Special Events and a default value of "BLOCKED" for all other traffic event types.	Decision Support	Configure Decision Support Use Case: ConfigureLaneClosureWord	Not Prototyped
SR4.2.3.2.10	The system shall recommend the use of pre-defined plans based on their applicability to the target	Decision Support		Screenshot: HMI Figure 13

Tag	Requirement	Feature	Use Cases	Other Design Elements
	traffic event.			
SR4.2.3.3.7.9	The system shall allow the user to add one or more DMS messages suggested by the decision support subsystem to the traffic event response plan.	Decision Support	Configure Decision Support Use Case: Add Suggested DMS Messages To Response Configure Decision Support Use Case: Add Suggested Plan Items To Response	Screenshot: HMI Figure 9 Screenshot: HMI Figure 10 Screenshot: HMI Figure 13 Screenshot: HMI Figure 14 ResponsePlanReqHdr:addSuggDMSsToResponsePlan SD
SR4.2.3.3.7.9.1	The system shall allow the user to indicate that the DMS message should be executed immediately when it is added to the traffic event response plan.	Decision Support	Configure Decision Support Use Case: Add Suggested DMS Messages To Response Configure Decision Support Use Case: Add Suggested Plan Items To Response	Screenshot: HMI Figure 9 Screenshot: HMI Figure 10 Screenshot: HMI Figure 13 Screenshot: HMI Figure 14
SR4.2.3.5.8	The system shall allow a user with the respond to traffic event right to view a traffic event response plan preview map	Decision Support	Configure Decision Support Use Case: View Response Plan Preview Map	Screenshot: HMI Figure 21 chartlite.servlet.trafficevents_classes CD ResponsePlanReqHdr:viewResponsePlanPreviewMap SD ResponsePlanReqHdr:getResponsePlanDecSupportData SD MapViewSpecificClasses CD ResponsePlanPreviewMap:initialize SD ResponsePlanPreviewMap:handleMapDataJSON SD MapReqHdr:getResponsePlanPreviewMapDataJSON SD
SR4.2.3.5.8.1	The preview map shall show the location of the traffic event.	Decision Support	Configure Decision Support Use Case: View Response Plan	Screenshot: HMI Figure 21

Tag	Requirement	Feature	Use Cases	Other Design Elements
			Preview Map	
SR4.2.3.5.8.2	The preview map shall show the location of each device in the traffic event response plan.	Decision Support	Configure Decision Support Use Case: View Response Plan Preview Map	Screenshot: HMI Figure 21
SR4.2.3.5.8.3	The user shall be able to click on any response device on the preview map and see the planned message for the device as defined in the response plan.	Decision Support	Configure Decision Support Use Case: View Response Plan Preview Map	Screenshot: HMI Figure 21
SR4.2.3.5.8.4	The system shall indicate if a device that is currently in the traffic event response plan should not be included based on decision support rules by visually differentiating the preview map marker for that device.	Decision Support	Configure Decision Support Use Case: View Response Plan Preview Map	Screenshot: HMI Figure 21
SR4.2.3.5.8.5	The system shall indicate if a device that is currently not in the traffic event response plan should be included based on decision support rules by visually differentiating the preview map marker for that device.	Decision Support	Configure Decision Support Use Case: View Response Plan Preview Map	Screenshot: HMI Figure 21
SR4.2.3.5.9	The system shall indicate if the response plan is currently missing a device that decision support rules indicate should be considered for response to the traffic event.	Decision Support	Configure Decision Support Use Case: View Devices That Should Be Considered for Response	Screenshot: HMI Figure 1 Screenshot: HMI Figure 15 chartlite.data.trafficevents_classes CD chartlite.servlet.trafficevents_classes CD
SR4.2.3.5.9.1	The system shall indicate if the response plan is currently missing a DMS device that is upstream, same direction and on the same route as a traffic event and is also within the	Decision Support	Configure Decision Support Use Case: View Devices That Should Be Considered for	Screenshot: HMI Figure 1 Screenshot: HMI Figure 15 Use Case: View Devices That Should Be Considered for Response

Tag	Requirement	Feature	Use Cases	Other Design Elements
	configurable distance.		Response	
SR4.2.3.5.10	The system shall indicate if a device is currently in a traffic event response plan that decision support rules indicate should not be there.	Decision Support	Use Case: View Device Use Warnings	Screenshot: HMI Figure 1 Screenshot: HMI Figure 19 chartlite.data.trafficevents_classes CD chartlite.servlet.trafficevents_classes CD
SR4.2.3.5.10.1	The system shall indicate if the response plan currently contains a DMS device that is downstream, same direction and on the same route as a traffic event.	Decision Support	Use Case: View Device Use Warnings	Screenshot: HMI Figure 1 Screenshot: HMI Figure 19
SR4.2.3.5.11	The system shall display the current active message on a device in the response plan if requested by the user.	Decision Support	Configure Decision Support Use Case: View Message Currently Active on Response Device	
SR4.2.3.5.11.1	The system shall display the current active message on a DMS device in the response plan if requested by the user.	Decision Support	Configure Decision Support Use Case: View Message Currently Active on Response Device	
SR4.3.6.1.3.5	The system shall allow a user with the respond to traffic event functional right to click on a link to request suggested response messages from the traffic event map popup.	Decision Support		
SR8.1.7	TSS traffic data shall be removed from the online system as soon as new traffic data is retrieved from the traffic sensor to replace it.	TSS	(existing fcn now incl G4)	(existing fcn now incl G4)
SR8.1.7.1	TSS traffic data removed from the online system shall be moved to an archive on the next scheduled	TSS	Archive TSS Data	(to be performed by DB archive job)

Tag	Requirement	Feature	Use Cases	Other Design Elements
	archive operation.			

## 12Acronyms/Glossary

<b>GIS</b>	<b>Geographic Information System</b> (GIS) is any system that captures, stores, analyzes, manages, and presents data that are linked to location
<b>Home Page Map</b>	The map component shown on the home page of the CHART user interface.
<b>Integrated Map</b>	The mapping components that are part of the CHART user interface.
<b>Intranet Map</b>	The CHART Mapping application that is not integrated into the CHART user interface.
<b>Location Alias</b>	A pre-defined location (lat/lon) that has been stored with some name attributes to allow operators to utilize the location repeatedly.
<b>Maintenance Portal</b>	A customized version of the CHART GUI tailored to device maintenance personnel.
<b>Nearby Devices Map</b>	Map shown on the details page for a traffic event that shows only the target traffic event and the devices that are near it.
<b>NTCIP</b>	National Transportation Communications for ITS Protocol. A family of standards designed to achieve interoperability and interchangeability between computers and electronic traffic control equipment from different manufacturers.
<b>Object Location Map</b>	Map component that is used in conjunction with the object location form when setting the location of a traffic event or device.
<b>Open Layers</b>	Open source JavaScript mapping API utilized by the integrated map components in the CHART GUI.
<b>REST</b>	Representational State Transfer - a web services architecture style used in CHART that leverages web technologies such as http and XML
<b>RWIS</b>	Roadway Weather Information System
<b>Standard GUI</b>	The CHART GUI when <i>not</i> accessed via the maintenance portal.
<b>TSS</b>	<b>Transportation Sensor System</b>
<b>WMS</b>	A Web Map Service (WMS) is a standard protocol for serving georeferenced map images over the Internet that are generated by a map server using data from a GIS database.